Journal of Artificial Intelligence Research 61 (2018) 593-621 Submitted 05/17; published 03/18

Fully Observable Non-deterministic Planning as Assumption-Based Reactive Synthesis

Nicolás D'Ippolito NDIPPOLITO@DC.UBA.AR
Instituto de Ciencias de la Computación
CONICET, Argentina

Natalia Rodríguez NRODRIGUEZ@DC.UBA.AR
Departamento de Computación, FCEN
Universidad de Buenos Aires, Argentina

Sebastian Sardina SEBASTIAN.SARDINA@RMIT.EDU.AU
School of Science (Computer Science)
RMIT University, Australia

Abstract

We contribute to recent efforts in relating two approaches to automatic synthesis, namely, automated planning and discrete reactive synthesis. First, we develop a *declarative* characterization of the standard "fairness" assumption on environments in non-deterministic planning, and show that strong-cyclic plans are *correct* solution concepts for fair environments. This complements, and arguably completes, the existing foundational work on non-deterministic planning, which focuses on characterizing (and computing) plans enjoying special "structural" properties, namely loopy but closed policy structures. Second, we provide an encoding suitable for reactive synthesis that avoids the naive exponential state space blowup. To do so, special care has to be taken to specify the fairness assumption on the environment in a succinct manner.

1. Introduction

The overarching aim of this work is to contribute to the recent efforts (e.g., Camacho, Triantafillou, Muise, Baier, & McIlraith, 2016; De Giacomo & Vardi, 2015; Kissmann & Edelkamp, 2009; Patrizi, Lipovetzky, De Giacomo, & Geffner, 2011; Pistore & Vardi, 2007; Sardina & D'Ippolito, 2015; Torres & Baier, 2015), in relating advanced forms of automated planning (Geffner & Bonet, 2013) to the general long-standing Computer Science problem of automatic synthesis (Abadi, Lamport, & Wolper, 1989; Manna & Waldinger, 1987; Pnueli & Rosner, 1989b): *the problem of automatically building an operational piece of code from (high-level) user intent*. Relating these two fields has the benefit of facilitating principle generalizations of the planning problem on the one hand, and grounding formal synthesis approaches in practical ways to Knowledge Representation formalisms for action and change, on the other hand. In particular, our work contributes to these efforts by *(i)* developing a *declarative* specification of the relation between the assumptions on the environment and the objective of a plan which arguably complements/completes the seminal work of Pistore and Traverso (2001) on non-deterministic fully observable planning; and *(ii)* providing an encoding of the assumption (and goal) that is suitable for controller synthesis that avoids the naive state space blowup, thus achieving the complexity of the planning task at hand.[1]

1. This work is an extension of IJCAI'15 conference paper by the first and third authors (Sardina & D'Ippolito, 2015), who remained the main contributors to this journal article (authors are listed alphabetically).

Fully observable non-deterministic (FOND) planning is an extension of classical planning that aims to accommodate events outside the control of the agent (Daniele, Traverso, & Vardi, 2000). In FOND planning, actions are non-deterministic, in that their execution yields one of a set of possible effects, and this is outside the control of the executor. Once the effect has ensued, however, the agent is able to observe its outcome. Strong-cyclic plans—those that re-try until success is obtained—have arguably become the de-facto solution concept for FOND planning. Daniele et al. provided a first characterization of strong-cyclic plans in CTL, as those plans for which *"in any of its executions, it is always the case that the goal can be reached."*[2] Since then, several promising techniques and actual planning systems have emerged for solving FOND planning problems (e.g., Cimatti et al., 2003; Fu, Ng, Bastani, & Yen, 2011; Kuter, Nau, Reisner, & Goldman, 2008; Muise, McIlraith, & Beck, 2012; Ramirez & Sardina, 2014). Such techniques amount to *specialized*—clever though often involved—algorithms for constructing plans that are "loopy" and "closed," structural properties directly capturing Daniele et al.'s requirements.

The fact is that neither Daniele et al.'s (2000) original definition nor the various algorithms for FOND planning make explicit the intended meaning of the solution concept, namely, that *strong-cyclic plans are those that bring about the goal when executed in "fair" environments.* A fair environment is one in which "every action executed infinitely often will exhibit all its effects infinitely often." In fact, such concept has either been informally discussed or considered only semantically, at the meta-level (e.g., Cimatti et al., 2003; Geffner & Bonet, 2013; Muise, McIlraith, & Belle, 2014; Patrizi, Lipovetzky, & Geffner, 2013; Ramirez & Sardina, 2014).

In this work, we *ground* the intended meaning of strong-cyclic plan as solution concept for FOND planning via a *declarative* characterization (Section 3) within Daniele et al.'s (2000) foundational framework, thus arguably "completing" it. As part of the motivation, we show how the naive specification of fairness does not yield the intended meaning, as it is not able to properly capture *effects' independence* (i.e., effects of different actions are *not coordinated* by any agent). Technically, we develop a logical specification of the environment assumptions for FOND planning (Definition 6), which we refer as *state-strong fair*, and prove that strong-cyclic plans, as defined by Daniele et al. (2000) and Pistore and Traverso (2001), are indeed sound and complete solution concepts under such assumptions (Theorem 3 and Corollary 1). We argue that such a declarative specification of the assumptions on the environment *(i)* avoids imprecise interpretations of the fairness conditions required for strong-cyclic plans to work; *(ii)* formally grounds the accepted intended meaning of the solution concept for FOND planning; and finally *(iii)* facilitates the statement of the planning problem as a general synthesis one, thus promoting cross-fertilization.

The declarative formalization of strong-cyclic plans can be leveraged to immediately express the FOND planning task as an instance of *controller (or reactive) synthesis* (Bloem, Jobstmann, Piterman, Pnueli, & Sa'ar, 2012; D'Ippolito, Braberman, Piterman, & Uchitel, 2013; Pnueli & Rosner, 1989a, 1989b). However, such direct restatement will rely on full LTL synthesis, which is known to be 2EXPTIME-Complete (Rosner, 1992), while FOND planning is EXPTIME (Rintanen, 2004). Thus, in the second part of the article (Section 4), we show how to specify the state-strong fair assumption in a *succinct* manner, by using a clever encoding that is inspired by that of Chatterjee,

2. Later further refined to disregard what happens after goal achievement (Cimatti, Pistore, Roveri, & Traverso, 2003; Pistore & Traverso, 2001).

Jurdzinski, and Henzinger (2004). Concretely, we develop a specification of FOND planning as a Büchi control problem that is optimal w.r.t. computational complexity.[3]

We believe that the above two developments contribute to a sharper understanding of non-deterministic planning and facilitate the synergies between the planning and the controller synthesis communities. Indeed, the planning community can take advantage of the expressiveness, rigorosity, and techniques provided by controller synthesis approaches. For example, one could "manipulate" the characterization of strong-cyclic proposed to capture more subtle concepts, such as domains with *conditional fairness* (i.e., where effects' fairness depend on the state); see Section 6. Also, reactive synthesis techniques, being exhaustive in nature, can yield "universal" type of plan solutions,[4] and while deemed computationally impossible in the past, recent approaches (e.g., Bloem et al., 2012; D'Ippolito, Braberman, Piterman, & Uchitel, 2011) have shown that for some quite expressive specifications, the task is amenable for computation. In turn, planning formalisms can inform meaningful specifications—for goals and assumptions—to the reactive synthesis field. Also works like the one shown here can facilitate researchers in the synthesis community to import computational techniques from the planning field, as it is the case whereby a (domain-independent) heuristic search is used to efficiently explore the solution space of reactive controllers (Ciolek, Braberman, D'Ippolito, & Uchitel, 2016).

Before diving into the technical details, we point out that achieving a specification for FOND planning capturing the fairness assumption explicitly that is *declarative* is central to this work. Of course analogous definitions and results could be obtained differently, in alternative formalisms. Particularly, there is an implicit connection between fairness and non-zero probabilities of uncontrollable events, and between strong-cyclic solution plans and success probability of 1 (Forejt, Kwiatkowska, Norman, & Parker, 2011). As a matter of fact, our encoding in Section 4 is itself inspired on work in *stochastic* reasoning (Chatterjee et al., 2004). Our approach, tough, does not require any probabilistic infrastructure and can be integrated within Daniele et al.'s (2000) foundational framework for FOND planning in a straightforward manner. We believe, still, that while understood by the community at a conceptual level, the links between logic-based declarative specifications and probabilistic properties are worth a precise formalization (but out of the scope of this work; see Section 5 for further discussion).

2. Preliminaries

We review the background on non-deterministic planning, CTL and LTL temporal logics, and reactive synthesis required to understand the technical development of the following sections.

2.1 Fully Observable Non–Deterministic Planning

We mostly follow the characterisation of non-deterministic given by Rintanen (2003; 2008), which captures the usual "*oneof*" clauses in PDDL representations (Gerevini, Bonet, & Givan, 2006).

A ***fully observable non-deterministic (FOND) planning problem*** is a tuple $\mathcal{P} = \langle P, O, s_I, \phi_{\text{goal}} \rangle$ consisting of a set of Boolean state propositions P (atoms), an initial state s_I, a goal ϕ_{goal} as a conjunction of literals (i.e., atoms or negated atoms), and an operator set O (see below). We use $\bar{l}$ to

3. Sardina and D'Ippolito (2015) showed that for the special case of FOND planning in which actions have "intended effects," existing efficient controller synthesis techniques developed in the Software Engineering community (D'Ippolito et al., 2013) can be exploited. The results in the current paper generalize such a special case.

4. That is, they generally build not one solution, but a family of solutions, often *all* solutions.

denote the complement of literal l. A ***state*** s is a consistent set (or conjunction) of literals such that $|s| = |P|$ — every atom is either true or false. We use S to denote the set of all states of task $\mathcal{P}$.

An ***operator*** is a pair $o = \langle Pre_o, \mathit{Eff}_o \rangle$, where Pre_o is a condition describing the *preconditions* of operator o, and $\mathit{Eff}_o = e_1 \mid \cdots \mid e_n$ the (non-deterministic) *effects* of o, where each e_i is a (deterministic effect) condition and $n \geq 1$. The intended meaning is that *one* of the e_i events ensue non-deterministically, by the environment's choice. Operator o is ***executable*** on a state s if $s \models Pre_o$, and the set of possible ***successor*** states from its execution is $\mathit{next}(o, s) = [\![\mathit{Eff}_o]\!]_s$, where:

$$[\![e_1 \mid \cdots \mid e_n]\!]_s = \bigcup_{i=1}^{n} \{(s \setminus \{\bar{l} \mid e_i \models l\}) \cup \{l \mid e_i \models l\}\}.$$

So, set $\mathit{next}(o, s)$ denotes those states that the world may evolve to when operator o is performed in state s. If o is deterministic (i.e., $\mathit{Eff}_o = e$), then $|[\![\mathit{Eff}_o]\!]_s| = 1$, for every state $s \in S$.

A ***policy*** (or *conditional plan*) is a function $\pi : S \mapsto 2^O$ mapping states $s \in S$ onto the set of executable operators $\pi(s)$ such that if $o \in \pi(s)$, then $s \models Pre_o$. The ***universal policy*** for a FOND problem $\mathcal{P}$ is $\hat{\pi}(s) = \{o \mid o \in O, s \models Pre_o\}$. A policy π is ***deterministic*** if $|\pi(s)| \leq 1$, for all $s \in S$. When π is deterministic we write $\pi(s) = o$ to compactly denote $\pi(s) = \{o\}$. Note that the non-determinism embodied in a policy (when $|\pi(s)| > 1$) is of different nature to that of actions' executions, as it is the agent, not nature, who decides which of the actions to perform. A policy π executed from state s defines a set of ***possible executions*** $\Lambda_\pi(s)$ made up of executions $\lambda = s_0 o_0 s_1 \cdots s_i o_i s_{i+1} \cdots$, where $s_0 = s$, $o_i \in \pi(s_i)$, $s_i \models Pre_{o_i}$, and $s_{i+1} \in \mathit{next}(o_i, s_i)$, for all $i \geq 0$. The set of states ***relevant*** to a policy π from state s is defined as $S_\pi(s) = \bigcup_{\lambda \in \Lambda_\pi(s)} \{s_i \mid s_i \in \lambda\}$ (we abuse notation and write $s \in \lambda$ to say that execution λ mentions state s). A policy π is ***closed*** iff $\bigcup_{o \in \pi(s_i)} \mathit{next}(o, s_i) \subseteq S_\pi(s_I)$ for all states $s_i \in S_\pi(s_I)$, that is, π always returns an action for every non-goal state it reaches. In turn, a state s is ***reachable*** by a policy π from state s' if there is a chance that following π leads the agent to s; formally, there exists $\lambda \in S_\pi(s')$ such that $s \in \lambda$.

When it comes to FOND planning, the usual solution concept in the literature is that of strong-cyclic plans.

Definition 1 (Bryce & Buffet, 2008; Cimatti et al., 2003). A policy π is a ***strong-cyclic plan*** for a planning problem if π is a closed policy such that a goal state is reachable from every reachable state using the policy, or equivalently, all partial executions of π can be extended to a finite execution path of π to a goal state. ■

A strong-cyclic plan guarantees, under plausible assumptions, that the agent eventually does achieve the goal.[5] What exactly those assumptions are and how they can be effectively captured declaratively is at center of this work.

We close by noting that while classical deterministic planning is PSPACE-complete (Bylander, 1994), non-deterministic planning with full observability is EXPTIME-complete (Rintanen, 2004).

2.2 Temporal Logics

Temporal logics allow the formal specification of behavioral properties of systems over time (Pnueli, 1977) and are the basis for model checking and synthesis. Some widely used and studied such logics

5. *Strong* policies are a special case for which all executions are finite and *acyclic*: they solve the planning problem in a bounded number of steps.

$\mathsf{CTL}^{\star}$, and its fragments CTL (Computation Tree Logic) and LTL (Lineal Temporal Logics) (Clarke & Emerson, 1982; Emerson & Halpern, 1986; Pnueli, 1977). Whereas CTL is the basis behind the foundational work in non-deterministic planning (Daniele et al., 2000; Pistore & Traverso, 2001), LTL is the basis for much work on reactive synthesis. The syntax of $\mathsf{CTL}^{\star}$, which subsumes both, is defined with the following grammar:

$$\Psi ::= p \mid \Psi_1 \wedge \Psi_1 \mid \neg\Psi \mid \mathsf{A}\varphi \mid \mathsf{E}\varphi;$$
$$\varphi ::= \Psi \mid \varphi_1 \wedge \varphi_1 \mid \neg\varphi \mid \mathsf{X}\varphi \mid \mathsf{F}\varphi \mid \mathsf{G}\varphi \mid \varphi_1 \mathsf{U} \varphi_2 \mid \varphi_1 \mathsf{W} \varphi_2,$$

where p ranges over the set of propositions. Formulas of the form Ψ are called state formulas, whereas those of the form φ are said to be path formulas. Formula $\mathsf{A}\Psi$ ($\mathsf{E}\Psi$) states that all executions (some execution) from the current state satisfy property Ψ. Then, path formulas $\mathsf{X}\varphi$, $\mathsf{F}\varphi$, and $\mathsf{G}\varphi$ state that φ is true in the next state of the path, eventually in the path, or always along the path, resp. Finally, $\varphi_1 \mathsf{U} \varphi_2$ says that φ_1 holds along the path until φ_2 becomes true and φ_2 is eventually true; $\varphi_1 \mathsf{W} \varphi_2$ is its weak version where φ_2 is not required to eventually come true in the path. Common combinations of path and state quantifiers allow us to say things be like *all* or *some* next states satisfy φ ($\mathsf{AX}\varphi$ and $\mathsf{EX}\varphi$), and φ holds in all or some executions ($\mathsf{AG}\varphi$ or $\mathsf{EG}\varphi$).

The fragment CTL of $\mathsf{CTL}^{\star}$ is obtained by requiring that each temporal path quantifiers (e.g., X, F, etc.) be under the immediate scope of an A or E quantifier. So, $\mathsf{EF}\varphi$ ("there exists an execution in which φ eventually holds") and $\mathsf{AG}\varphi$ ("φ always holds along all executions") are legal CTL formulas, but $\mathsf{AGF}\varphi$ ("φ is true infinitely often in every execution path") is not. In turn, LTL is the fragment obtained by withholding the A and E quantifiers and assuming universal path quantification. So, non-CTL formula $\mathsf{A}(\mathsf{GF})\phi$ can in fact be captured in LTL, but formulas examining alternative executions along a path, such as $\mathsf{AGE}\phi$ ("ϕ is always potentially reachable"), cannot.

The meaning of $\mathsf{CTL}^{\star}$ formulas (and of its fragments CTL and LTL) is given over the states and paths of a transition system, with a branching-time interpretation of time. Concretely, a $\mathsf{CTL}^{\star}$ transition system over a set of propositions P, also called Kripke structure (Emerson, 1990), is a tuple $\mathcal{K} = \langle W, R, P \rangle$, where:

- $W \subseteq 2^P$ is the set of states of $\mathcal{K}$; and
- $R \subseteq W \times W$ is the transition relation of $\mathcal{K}$. When $R(w_1, w_2)$ holds, it means that state w_2 is a possible successor of state w_1.

A ***run*** in $\mathcal{K}$ from state w_0 is a, possibly infinite, sequence $\lambda = w_0 w_1 \cdots$ such that $R(w_i, w_{i+1})$, for all $i \geq 0$. A ***maximal run*** is either an infinite run or a finite run that ends in a terminal state, that is, a state that has no possible successor state in $\mathcal{K}$.[6]

Then, given a state formula Ψ and a state $w \in W$, one can define whether Ψ holds true (i.e., is satisfied) in structure $\mathcal{K}$ at state w, denoted $\mathcal{K}, w \models \Psi$, structurally on the formula and resorting on runs. For example, for CTL we have $\mathcal{K}, w \models p$ *iff* $w \models p$; and $\mathcal{K}, w \models \mathsf{EF}\phi$ *iff* there exists a maximal run $\lambda = w_0 w_1 w_2 \cdots$ with $w_0 = w$ such that for some $k \geq 0$, $\mathcal{K}, w_k \models \phi$). Similarly, the semantics of LTL formulas is defined in terms of (infinite) runs of $\mathcal{K}$: $\mathcal{K}, w \models \varphi$ *iff* $\lambda \models \varphi$, for all infinite runs $\lambda = w w_1 \cdots$ of $\mathcal{K}$ (that is, φ holds true in all runs starting in state w), where *(i)* $\lambda \models p$ iff $w_0 \models p$; *(ii)* $\lambda \models \neg p$ iff $w_0 \models \neg p$; *(iii)* $\lambda \models \varphi_1 \wedge \varphi_2$ iff $\lambda \models \varphi$ and $\lambda \models \varphi_2$; *(iv)* $\lambda \models \mathsf{X}\varphi$ iff $w_1 w_2 \cdots \models \varphi$; and *(v)* $\lambda \models \varphi_1 \mathsf{U} \varphi_2$ iff there exists $j \geq 0$ such that $w_j \models \varphi_2$ and, for all $0 \leq k < j$,

6. Observe that, unlike some definitions of CTL, we account, wlog, for structures that include terminal states.

$w_k \models \varphi_1$. As usual, we write $\mathcal{K} \models \phi$ to denote $\mathcal{K}, w \models \phi$ for all $w \in W$. See Baier and Katoen's (2008) book for more details on these logics and their use in model checking.

2.3 Reactive Synthesis on Two-Player Games

First introduced by Church (1963), automatic synthesis of programs is one of the most ambitious problems in Computer Science. Intuitively, it involves the automatic construction of an operational model for a so-called *controller* that, deployed in a (model of the) *environment* results in a system that is guaranteed to satisfy a given *goal*.

The problem was considered by Pnueli and Rosner (1989a, 1989b) in the context of synthesizing *reactive* modules from a specification given in LTL (Pnueli, 1977; Vardi, 1996). There, both the environment and the system goal are specified by LTL formulas, and the task is to produce an operational specification of a module that restricts the traces allowed by the environment to only those satisfying the LTL system goal. Technically, the set of propositions describing the domain of concern is partitioned into two disjoint sets $\mathcal{X}$ (the non controllable propositions) and $\mathcal{Y}$ (the controllable propositions). The problem then is: *can we control the values of $\mathcal{Y}$ such that for every possible value of $\mathcal{X}$ for every behaviour of the module a certain LTL formula φ holds?* More precisely, runs (c.f. Section 2.2) can be interpreted as of the form $\lambda = (X_0, Y_0)(X_1, Y_1)(X_2, Y_2)\cdots$, where (X_i, Y_i) is the propositional interpretation at the i-th position in λ, now partitioned in the propositional interpretation X_i for $\mathcal{X}$ and Y_i for $\mathcal{Y}$. Let us denote by $\lambda_{\mathcal{X}}|_i$ the run λ projected only on $\mathcal{X}$ and truncated at the i-th element (included), i.e., $\lambda_{\mathcal{X}}|_i = X_0 X_1 \cdots X_i$. The ***realizability problem*** checks the existence of a function $f : (2^{\mathcal{X}})^* \rightarrow 2^{\mathcal{Y}}$ such that for all λ with $Y_i = f(\lambda_{\mathcal{X}}|_i)$ we have that λ satisfies the formula φ (i.e., $\lambda \models \varphi$). The (reactive) ***synthesis problem*** consists in actually computing such a function. Observe that in realizability/synthesis, there is no way to constrain the value assumed by the propositions in $\mathcal{X}$: the function we seek *only* acts on propositions in $\mathcal{Y}$.

A popular approach to solving the synthesis problem is by visualizing it as the solution of a two-player antagonistic game (D'Ippolito et al., 2013; Piterman, Pnueli, & Sa'ar, 2006; Pnueli, 1977; Vardi, 1996) (originally introduced in von Neumann, Morgenstern, Kuhn, & Rubinstein, 1944), in which controller and environment are considered adversarial players, and the winning condition represents the LTL goal specification. For this paper, it is enough to consider *simple games*, that is, non-terminating turn-based zero-sum perfect-information two-player games played over a finite state graph (S, E, s_0) between two players $\Box$ and $\Diamond$, who move a token by turns from state to state along edges into the graph so that an infinite path is formed. Non-terminating zero-sum games (with ω-regular winning conditions) are determined, which means that from each state one of the two players wins and the other loses. In perfect-information games, all participants have all the required information to make the decision to make the game move. We start by defining the space where the game takes place.

Definition 2. A ***2-player game graph*** is a tuple of the form $\mathcal{G} = \langle (S, E, s_0), (S_\Box, S_\Diamond) \rangle$, where (S, E, s_0) is a finite directed graph with state set S, edge set E, and initial state $s_0 \in S$, and $(S_\Box, S_\Diamond)$ is a partition of the state set S, denoting the states in which players $\Box$ and $\Diamond$ chooses the successor states, respectively. ■

Plays A ***play*** $\rho = s_0 s_1 s_2 \cdots$ over a game graph $\mathcal{G}$ is an infinite sequence of states in S starting from $\mathcal{G}$'s initial state s_0 and such that $(s_i, s_{i+1}) \in E$, for all $i \geq 0$. We write Ω for the set of all plays, Ω_s for the set of all plays starting from the state s, and $\Delta_{\mathcal{G}}(s)$ for the set of all states s' such that $(s, s') \in E$ (i.e., all possible successors of s). We denote $\rho[i] = s_i$ the ith state in a play ρ.

Strategies The choices of players are formalized in the form of a ***strategy***, which is the policy that the player applies to move along the game graph. Although in general the strategy that a player "uses" or "follows" in the game may depend, or not, on the history of a play, in the context of this work we focus only in *memoryless* strategies. Even more, we are interested in (*pure*) strategies in which only one successor can be selected to pass the token to.

Formally, such a strategy for $\Box$ is a function $\sigma_\Box : S_\Box \to S$, whereas a strategy for $\Diamond$ is a function $\sigma_\Diamond : S_\Diamond \to S$. Player $\Box$ follows (or uses) a strategy $\sigma_\Box$ if in each move, at a state $s \in S_\Box$ she chooses a successor state according to $\sigma_\Box(s)$. A play $\rho = s_0, s_1, \ldots$ is ***consistent*** with a strategy $\sigma_\Box$ if for every $i \geq 0$ such that $s_i \in S_\Box$, $\sigma_\Box(s_i) = s_{i+1}$. Consistency with respect to $\sigma_\Diamond$ is defined analogously. We write $\Sigma_\Box$ and $\Sigma_\Diamond$ for the set of all strategies for $\Box$ and $\Diamond$ respectively.

Once strategies $\sigma_\Box \in \Sigma_\Box$ and $\sigma_\Diamond \in \Sigma_\Diamond$ are fixed, a play starting in state $s \in S$ is determined by an infinite path (denoted $\rho_s^{\sigma_\Box, \sigma_\Diamond}$), consistent with $\sigma_\Box$ and $\sigma_\Diamond$. We define $\mathcal{G}_{\sigma_\Box, \sigma_\Diamond}$ as the sub-graph obtained by pruning $\mathcal{G}$ according to the decisions made by strategies $\sigma_\Box$ and $\sigma_\Diamond$.

Winning conditions Winning conditions of players in simple games are specified by a set of ***winning plays*** $\mathcal{W} \subseteq \Omega$ for a player. When the set of states S is modelled via a factored representations (i.e., using propositional variables), the set of plays can be seen as LTL runs and the winning plays as LTL formulas (e.g., all the runs where p is eventually true).

As we consider zero-sum 2-player games if the winning condition of a player is the set $\mathcal{W}$, then the winning condition for the other player is $\Omega \backslash \mathcal{W}$. Given a game graph $\mathcal{G}$ and a winning condition $\mathcal{W}$, we write $\mathcal{G}(\mathcal{W})$ for the game played on $\mathcal{G}$ with winning condition $\mathcal{W}$ for $\Box$. We say that a strategy $\sigma_\Box \in \Sigma_\Box$ is a ***sure winning*** strategy for $\Box$ in the game $\mathcal{G}(\mathcal{W})$ if, for all $\sigma_\Diamond \in \Sigma_\Diamond$ we have $\rho^{\sigma_\Box, \sigma_\Diamond} \in \mathcal{W}$. In words, a sure winning strategy is one that guarantees to produce a winning play, no matter how the other player happens to play. The set of states from which $\Box$ (resp. $\Diamond$) can surely win (called ***winning set***) is denoted as $\mathcal{W}_\Box$ (resp., $\mathcal{W}_\Diamond$). A game $\mathcal{G}(\mathcal{W})$ is ***determined*** if for all state in S, either $\Box$ or $\Diamond$ can surely win from such state.

In this work we will focus on a specific type of winning conditions called Büchi conditions. Büchi conditions are given as LTL formulas of the form $\mathsf{GF}\varphi$ (Buchi & Landweber, 1969). Thus, traces satisfying a Büchi condition $\mathsf{GF}\varphi$ are those that satisfy φ infinitely many times.

Finally, realizability and synthesis in the context of two-player games amount to checking the existence of and computing, resp., a sure winning strategy for the player of interest.

3. Strong Cyclic Plans and Fair Environments

In this section, we revisit the semantics of FOND planning, with special focus on the "environment" where plans are to be executed. In particular, whereas the notion of strong-cyclic plans have been vastly studied and formally defined, the type of environments in which such plans will succeed are often discussed at an informal level.

The first precise analysis of what a solution is for FOND planning was given by Daniele et al. (2000) and Pistore and Traverso (2001). There, the authors formally defined strong-cyclic plans through a CTL formula on their executions. To capture all executions of a policy, they defined what is basically the projection of the underlying state model for a planning task onto the policy. For the rest of the paper, we follow Daniele et al. and, wlog, restrict our attention to FOND planning problems with *no dead-end states* (i.e., in every state there is always some executable operator), and policies that are *closed* (i.e., those that always specify how to proceed for all the possible outcomes of any action in the policy).[7]

Definition 3 (($\mathcal{P}, \pi$)-structure). Let $\mathcal{P} = \langle P, O, s_I, \phi_g \rangle$ be a FOND planning problem over the set of propositions $\mathcal{P}$ and π a policy over $\mathcal{P}$. The induced structure of $(\mathcal{P}, \pi)$ is a Kripke structure $\mathcal{K}^{\pi}_{\mathcal{P}} = \langle W, R, P \rangle$, where:

- $W = \{\langle s, o \rangle \mid s \in 2^P, o \in O\}$. Intuitively, $\langle s, o \rangle$ represents the execution of operator o in state s; and
- $R(\langle s, o \rangle, \langle s', o' \rangle)$ iff $o \in \pi(s)$, $o' \in \pi(s')$, and $s' \in \textit{next}(o, s)$. ■

Structure $\mathcal{K}^{\pi}_{\mathcal{P}}$ represents all evolutions of policy π when executed in planning domain $\mathcal{P}$. It is straightforward to define a structure representing all possible executions of the planning domain, by considering the universal policy.

Definition 4 ($\mathcal{P}$-structure). The structure induced by a FOND planning problem $\mathcal{P}$ is defined as the Kripke structure $\mathcal{K}_{\mathcal{P}} = \mathcal{K}^{\hat{\pi}}_{\mathcal{P}}$, where $\hat{\pi}$ stands for the universal policy. ■

So, Daniele et al. (2000) defined a policy π as a strong-cyclic solution for a planning problem $\mathcal{P}$ iff $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s_I, o_I \rangle \models \mathsf{AGEF}\phi_{\text{goal}}$, for all $o_I \in \pi(s_I)$. In words, starting from the initial state, whatever actions we choose to execute (from the policy) and whatever their outcomes are, we always (AG) have a way of reaching the goal ($\mathsf{EF}\phi_{\text{goal}}$).

Because goals are *achievement* goals in planning, what happens after the goal is achieved is irrelevant. Hence, Daniele et al.'s definition was then further refined as follows:

Definition 5 (Pistore & Traverso, 2001). A policy π is a ***strong-cyclic plan solution*** for a FOND planning problem $\mathcal{P}$ iff $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s_I, o_I \rangle \models \mathsf{A}(\mathsf{EF}\phi_{\text{goal}} \mathsf{W} \phi_{\text{goal}})$, for all $o_I \in \pi(s_I)$. ■

In words, in all possible executions of the policy, the goal is always eventually reachable, at least until the goal is indeed reached. In other words, wherever the policy may take us, the goal will be (potentially) reachable from there. Observe that this definition precisely captures the notion of closeness and goal-reachable (as in Definition 1).

The availability of a precise notion for strong-cyclic plans has facilitated the development of various techniques capable of synthesising such type of plans. Planning approaches and tools such as MBP (Cimatti et al., 2003), PRP (Muise et al., 2012), GRENDEL (Ramirez & Sardina, 2014), myND (Ortlieb & Mattmüller, 2013), GAMER (Kissmann & Edelkamp, 2009), and FIP (Fu et al., 2011) all search for policies that are closed and goal-reachable.

Nevertheless, besides understanding what a strong-cyclic plan amounts to, we argue here that it is also important to understand and formalize *the contexts under which these type of plans will indeed achieve the objectives*, namely, bringing about the goal. This has not received much attention and has mostly been discussed informally.

Clearly, under non-determinism, there is in principle the possibility of never achieving the goal, as "strong-cyclic solutions can produce executions that loop forever" (without ever reaching the

7. As customary in verification, these assumptions can be easily met by adding "no-op" actions in dead-end states. All results in the paper apply also to planning problems with dead-ends and/or partial policies; one just needs to consider finite execution traces as special cases.

goal) (Cimatti et al., 2003). However, the common understanding in the literature is that strong-cyclic plans are adequate solutions under the assumption that the underlying environment described by the planning domain is *"fair."* Unlike strong-cyclic plans, which have been formally characterized (see above), this fairness assumption is always stated at an informal level. We shall provide a precise characterization of such fairness in LTL and show that strong-cyclic plans are sound and complete plan types for fair domains.

A usual (informal) understanding is that fair domains are those for which *"if an action is executed infinitely many times, every non-deterministic outcome will occur infinitely often."* Formally, this can be stated as the following strong-fairness LTL constraint (on the induced structure $\mathcal{K}_{\mathcal{P}}$):

$$\Phi_{\mathcal{P}}^{\mathit{fair}} \stackrel{\text{def}}{=} \bigwedge_{o \in O, e \in \mathit{Eff}_o} (\mathsf{GF}o \rightarrow \mathsf{GF}e).$$

Note that while strong-cyclic plans have been characterized in CTL (Definition 5), formula $\Phi_{\mathcal{P}}^{\mathit{fair}}$ above is an LTL formula not expressible in CTL. While simple, D'Ippolito et al. (2011) argued, in the context of reactive synthesis for Software Engineering, that such strong-fairness assumptions are not enough to guarantee the success of controllers that are meant to "re-try." Consider the following counter example.[8]

Example 1. Consider the standard problem of obtaining two heads on the table by flipping two (fair) coins. To flip a coin, the agent needs to be holding them; to that end, she has an action to pick up all the coins on the table. Flipping a coin may yield, non-deterministically, the coin with heads or tails (not heads) on the table. If in a trial, the agent did not obtain two heads, it needs to pick them up and start all over again. Thus, consider planning problem $\mathcal{P}_{\mathit{coin}} = \langle P, O, s_I, \phi_{\text{goal}} \rangle$, where:

- $P = \{\mathit{heads}_1, \mathit{heads}_2, \mathit{holding}_1, \mathit{holding}_2\}$.
- O includes the operators:
  - $\text{PICK} = \langle \neg \mathit{holding}_1 \wedge \neg \mathit{holding}_2, \mathit{holding}_1 \wedge \mathit{holding}_2 \wedge \neg \mathit{heads}_1 \wedge \neg \mathit{heads}_2 \rangle$.
  - $\text{FLIP}_i = \langle \mathit{holding}_i, (\mathit{heads}_i \wedge \neg \mathit{holding}_i) \mid (\neg \mathit{heads}_i \wedge \neg \mathit{holding}_i) \rangle$, for $i \in \{1, 2\}$.
- $s_I = \{\neg \mathit{holding}_1, \neg \mathit{holding}_2, \neg \mathit{heads}_1, \neg \mathit{heads}_2\}$.
- $\phi_{\text{goal}} = \mathit{heads}_1 \wedge \mathit{heads}_2 \wedge \neg \mathit{holding}_1 \wedge \neg \mathit{holding}_2$.

Take the following (deterministic) policy:

$$\pi(s) = \begin{cases} \text{PICK} & \text{if } s \models \neg \mathit{holding}_1 \wedge \neg \mathit{holding}_2 \\ \text{FLIP}_1 & \text{if } s \models \mathit{holding}_1 \wedge \mathit{holding}_2 \\ \text{FLIP}_2 & \text{if } s \models \neg \mathit{holding}_1 \wedge \mathit{holding}_2 \end{cases}$$

It is not hard to see that π is a strong-cyclic solution: it continuously tries to get both heads, by picking the coins and flipping them, until success is achieved. Observe that, in this example, it is assumed that it is not possible to just pick one of the coins, they have to be picked together.

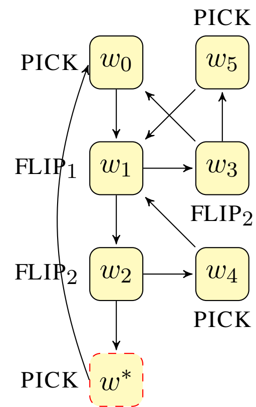
8. The car example provided by Sardina and D'Ippolito (2015) is significantly more involved.

Now, the formula $\Phi^{\mathit{fair}}_{\mathcal{P}_{coin}}$, as defined above, implies that if the agent tries to use the key infinitely often, then she will succeed infinitely often. Similarly, if the agent keeps flipping the coin, then she will keep succeeding in getting heads. It turns out, though, that:

$$\mathcal{K}^{\pi}_{\mathcal{P}_{coin}}, \langle s_I, \text{PICK}\rangle \not\models \mathsf{A}[\Phi^{\mathit{fair}}_{\mathcal{P}_{coin}} \rightarrow \mathsf{F}(\mathit{heads}_1 \wedge \mathit{heads}_2)].$$

That is, there is an execution of π in $\mathcal{P}$ in which all outcomes of, PICK, FLIP_1, and FLIP_2 arise infinitely often, but never two heads are obtained on the table. In fact, consider the following states of structure $\mathcal{K}^{\pi}_{\mathcal{P}_{coin}}$:

- $w_0 = \langle s_I, \text{PICK}\rangle$;
- $w_1 = \langle\{\mathit{holding}_1, \mathit{holding}_2, \neg\mathit{heads}_1, \neg\mathit{heads}_2\}, \text{FLIP}_1\rangle$;
- $w_2 = \langle\{\neg\mathit{holding}_1, \mathit{holding}_2, \mathit{heads}_1, \neg\mathit{heads}_2\}, \text{FLIP}_2\rangle$;
- $w_3 = \langle\{\neg\mathit{holding}_1, \mathit{holding}_2, \neg\mathit{heads}_1, \neg\mathit{heads}_2\}, \text{FLIP}_2\rangle$;
- $w_4 = \langle\{\neg\mathit{holding}_1, \neg\mathit{holding}_2, \mathit{heads}_1, \neg\mathit{heads}_2\}, \text{PICK}\rangle$; and
- $w_5 = \langle\{\neg\mathit{holding}_1, \neg\mathit{holding}_2, \neg\mathit{heads}_1, \mathit{heads}_2\}, \text{PICK}\rangle$.



Now consider run $\lambda = w_0(w_1w_2w_4w_1w_3w_5)^{\omega}$. It is clear that λ does satisfy the assumption formula $\Phi^{\mathit{fair}}_{\mathcal{P}_{coin}}$, but it *never reaches the goal*. The issue here is that the failure of the action FLIP_2 is *not* independent of failure of previous action FLIP_1—their failures are "synchronized." ■

The example above shows that the failures and successes of two non-deterministic actions can be "coordinated" in a way that will preclude goal achievement, despite the fact that all outcomes of both actions ensue infinitely often. To rule out coordinated outcomes, D'Ippolito et al. (2011) provided a stronger notion of fair environments, but their framework assumes that it is known which action outcomes are "good" (and which ones are considered "failures"). Still, the draw from their intuition to have action-outcome fairness in every state as follows.

Definition 6 (State Strong Fairness). Let $\mathcal{P}$ be a FOND planning problem and $\mathcal{K}_{\mathcal{P}} = \langle W, R, P\rangle$ its corresponding Kripke structure. The ***state-strong fair constraint*** is defined as the LTL formula

$$\gamma^{\mathit{sfair}}_{\mathcal{P}} \stackrel{\text{def}}{=} \bigwedge_{\{\langle s,o,e\rangle \mid \langle s,o\rangle \in W, e \in \mathit{Eff}_o\}} (\mathsf{GF}(s \wedge o) \rightarrow \mathsf{GF}[\![e]\!]_s).$$

A run of λ of $\mathcal{K}_{\mathcal{P}}$ is state-strong fair if $\lambda \models \gamma^{\mathit{sfair}}_{\mathcal{P}}$. ■

In words, $\gamma^{\mathit{sfair}}_{\mathcal{P}}$ says that if an operator is performed infinitely often *in the same state*, then all its effects ought to arise infinitely often. This LTL constraint formalizes Cimatti et al.'s (2003) claim that unfair executions are those where "some actions are executed infinitely often *in given states*, but some of its outcomes never occur," and corresponds to the *semantically expressed* requirement used in various FOND planning works (e.g., Geffner & Bonet, 2013; Patrizi et al., 2013; Ramirez & Sardina, 2014).

Returning to Example 1, the run λ does *not* satisfy $\gamma^{\mathit{sfair}}_{\mathcal{P}_{coin}}$ (i.e., $\lambda \models \neg\gamma^{\mathit{sfair}}_{\mathcal{P}_{coin}}$), and hence $\lambda \models \gamma^{\mathit{sfair}}_{\mathcal{P}_{coin}} \rightarrow \mathsf{F}\phi_{\text{goal}}$ applies—run λ is unfair and shall not be taken into consideration. It is not difficult to check that indeed $\mathcal{K}^{\pi}_{\mathcal{P}} \models \mathsf{A}[\gamma^{\mathit{sfair}}_{\mathcal{P}_{coin}} \rightarrow \mathsf{F}\phi_{\text{goal}}]$ holds for such an example.

We now have all the machinery to state the main contributions of this section, namely, linking state-strong fair with Pistore and Traverso's (2001) account of strong-cyclic plans (see Definition 5). The following two main results restrict, wlog, to *deterministic* policies; we will discuss the impact of non-deterministic policies at the end of the paper (Section 6). First, we show that all such plans do achieve the goal in state-strong fair environments.

Theorem 1. Let π be a deterministic strong-cyclic plan for a FOND planning problem $\mathcal{P}$. Then, $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s_I, o_I\rangle \models \mathsf{A}[\gamma^{\mathit{sfair}}_{\mathcal{P}} \rightarrow \mathsf{F}\phi_{\text{goal}}]$, with $o_I = \pi(s_I)$.

Proof. Consider a run $\lambda = \langle s_0, o_0\rangle\langle s_1, o_1\rangle \cdots$ of Kripke structure $\mathcal{K}^{\pi}_{\mathcal{P}}$ where $s_0 = s_I$ and $o_0 = \pi(s_I)$. Suppose further that $\lambda \models \gamma^{\mathit{sfair}}_{\mathcal{P}}$. We aim to prove then that $\lambda \models \mathsf{F}\phi_{\text{goal}}$.

We first show the following intermediate claim: if there exists state $\langle \hat{s}, \hat{o}\rangle$ that is mentioned in λ infinite many times (i.e., $\lambda \models \mathsf{GF}(\hat{s} \wedge \hat{o})$) and from where a state in which ϕ_{goal} holds can be reached in $\mathcal{K}^{\pi}_{\mathcal{P}}$ (i.e., $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle \hat{s}, \hat{o}\rangle \models \mathsf{EF}\phi_{\text{goal}}$), then $\lambda \models \mathsf{F}\phi_{\text{goal}}$. Because $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle \hat{s}, \hat{o}\rangle \models \mathsf{EF}\phi_{\text{goal}}$, there is a finite trace $\tau = \langle s^0, o^0\rangle \cdots \langle s^n, o^n\rangle$, with $n \geq 1$, in structure $\mathcal{K}^{\pi}_{\mathcal{P}}$ such that $s^0 = \hat{s}$, $o^0 = \hat{o}$, and $s^n \models \phi_{\text{goal}}$. We prove this by induction on the length n:

- If $n = 0$, then $\hat{s} \models \phi_{\text{goal}}$ and $\lambda \models \mathsf{F}\phi_{\text{goal}}$ follows trivially as $\langle \hat{s}, \hat{o}\rangle$ is mentioned in run λ.
- Now suppose that for all $n \leq k$, with $k \geq 1$, $\lambda \models \mathsf{F}\phi_{\text{goal}}$ applies (induction hypothesis). Suppose that $n = k + 1$ (which means that $n \geq 1$):
  - Due to $\mathcal{K}^{\pi}_{\mathcal{P}}$ construction, $s^1 \in [\![e]\!]_{\hat{s}}$ for some effect $e \in \mathit{Eff}_{\hat{o}}$. Because $\lambda \models \gamma^{\mathit{sfair}}_{\mathcal{P}}$, it is the case that $\lambda \models \mathsf{GF}(\hat{s} \wedge \hat{o}) \rightarrow \mathsf{GF}s^1$. Since we assumed that $\lambda \models \mathsf{GF}(\hat{s} \wedge \hat{o})$, then we conclude $\lambda \models \mathsf{GF}s^1$. Intuitively, one of the effects e of operator $\hat{o}$ brings about goal state s^1 when applied in planning state $\hat{s}$, and operator $\hat{o}$ is indeed executed in planning state s_k during run λ infinitely often. Because λ is fair on all the effects of operator $\hat{o}$, such effect e ought to ensue at some point in λ, bringing about state s^1 infinite times.
  - So, we have that $\lambda \models \mathsf{GF}s^1$ and since τ represents an execution of π, it is the case that $\pi(s^1) = o^1$ and therefore $\lambda \models \mathsf{GF}(s^1 \wedge o^1)$, that is, $\langle s^1, o^1\rangle$ repeats infinitely often in λ. In addition, there exists trace $\tau' = \langle s^1, o^1\rangle \cdots \langle s^n, o^n\rangle$ of length k in structure $\mathcal{K}^{\pi}_{\mathcal{P}}$ achieving ϕ_{goal}; formally $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s^1, o^1\rangle \models \mathsf{EF}\phi_{\text{goal}}$. By induction hypothesis, we obtain then that $\lambda \models \mathsf{F}\phi_{\text{goal}}$.

Due to Definition 5 and the fact that π is a strong-cyclic plan, we know that:

$$\lambda \models \mathsf{EF}\phi_{\text{goal}}\mathsf{W}\phi_{\text{goal}}. \tag{1}$$

Suppose, by contradiction, that $\lambda \models \neg\mathsf{F}\phi_{\text{goal}}$. Then, due to 1, it follows that $\lambda \models \mathsf{GEF}\phi_{\text{goal}}$, that is, while λ never achieves ϕ_{goal}, goal ϕ_{goal} is potentially reachable from every step along λ. Since structure $\mathcal{K}^{\pi}_{\mathcal{P}}$ is finite, there must exist an $i \geq 0$ such that $\langle s_k, o_k\rangle$ (with $o_k = \pi(s_k)$) is visited infinitely often in λ, that is, $\lambda \models \mathsf{GF}(s_k \wedge o_k)$. Because $\lambda \models \mathsf{GEF}\phi_{\text{goal}}$ and $\langle s_k, o_k\rangle \in \lambda$, it is the case that $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s_k, o_k\rangle \models \mathsf{EF}\phi_{\text{goal}}$. Hence, we can apply the intermediate result we proved above to conclude that $\lambda \models \mathsf{F}\phi_{\text{goal}}$. Observe that equation 1 above applies because $\lambda \models \mathsf{EF}\phi_{\text{goal}}\mathsf{U}\phi_{\text{goal}}$ holds.

As λ is an run in $\mathcal{K}^{\pi}_{\mathcal{P}}$ starting in $\langle s_I, o_I\rangle$, with $o_I = \pi(o_I)$, then thesis follows. $\square$

This theorem provides a "soundness" result for state-strong fair environments. The theorem is related to one by Patrizi et al. (2013, Thm. 5), though ours crystallize the fairness assumption explicitly within Daniele et al.'s (2000) FOND planning logical foundational framework (see Section 5 for further discussion). The second, and less obvious, result states that if a plan guarantees the goal in a state-strong fair environment (i.e., an environment in which every possible trace is state-strong fair), then it has to be strong-cyclic.

Theorem 2. Let π be a deterministic policy and $\mathcal{P}$ a FOND problem such that $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s_I, o_I\rangle \models \mathsf{A}[\gamma^{\mathit{sfair}}_{\mathcal{P}} \rightarrow \mathsf{F}\phi_{\text{goal}}]$, for $o_I = \pi(s_I)$. Then, π is a strong-cyclic plan.

Proof. From Definition 5 we ought to show that $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s_I, \pi(s_I)\rangle \models \mathsf{A}(\mathsf{EF}\phi_{\text{goal}}\mathsf{W}\phi_{\text{goal}})$. Take a run λ in $\mathcal{K}^{\pi}_{\mathcal{P}}$ of the form $\lambda = \langle s^0, o^0\rangle\langle s^1, o^1\rangle \cdots$ such that $s^0 = s_I$ and $o^1 = \pi(s_I)$. We want to prove that $\lambda \models \mathsf{EF}\phi_{\text{goal}}\mathsf{W}\phi_{\text{goal}}$.

Take an arbitrary state $\langle s^k, o^k\rangle$ in λ, for some $k \geq 0$, such that for all $i \in \{0, \dots, k\}$, it is the case that $s^k \models \neg\phi_{\text{goal}}$. That is, suppose that up to $\langle s^k, o^k\rangle$, λ has not yet achieved the goal. We are to prove then that $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s^k, o^k\rangle \models \mathsf{EF}\phi_{\text{goal}}$, that is, the goal is reachable from $\langle s^k, o^k\rangle$.

Next, let $\lambda_k = \langle s^k, o^k\rangle\langle s^{k+1'}, o^{k+1'}\rangle\langle s^{k+2'}, o^{k+2'}\rangle \cdots$ be a run in $\mathcal{K}^{\pi}_{\mathcal{P}}$ from $\langle s^k, o^k\rangle$ that is fair, that is, such that $\lambda_k \models \gamma^{\mathit{sfair}}_{\mathcal{P}}$. It is easy to see that one such λ_k always exists and can easily be constructed by traversing structure $\mathcal{K}^{\pi}_{\mathcal{P}}$ in a way that every successor of a state that is mentioned infinitely often, appears infinitely often.

Next, consider the run λ' built from concatenating the first k steps of λ with λ_k, that is:

$$\lambda' = \langle s^0, o^0\rangle\langle s^1, o^1\rangle \cdots \langle s^k, o^k\rangle\langle s^{k+1'}, o^{k+1'}\rangle\langle s^{k+2'}, o^{k+2'}\rangle \cdots$$

Since $\lambda_k \models \gamma^{\mathit{sfair}}_{\mathcal{P}}$ and λ' is just λ_k modulo a finite prefix, it follows that $\lambda' \models \gamma^{\mathit{sfair}}_{\mathcal{P}}$, that is, run λ' itself is fair. Now, given that λ' is a run from $\langle s_I, \pi(s_I)\rangle$, then by the assumption in the theorem statement, $\lambda' \models \gamma^{\mathit{sfair}}_{\mathcal{P}} \rightarrow \mathsf{F}\phi_{\text{goal}}$ follows, which together with $\lambda' \models \gamma^{\mathit{sfair}}_{\mathcal{P}}$, implies that $\lambda' \models \mathsf{F}\phi_{\text{goal}}$.

From the fact that the first $k+1$ steps in λ' and λ are the same (i.e., from $\langle s^0, o^o\rangle$ to $\langle s^k, o^k\rangle$ included), and the fact that λ does not achieve the goal up to step $k+1$ included (see above), it follows that λ' does not achieve the goal in the first $k+1$ steps either. However, we have shown that run λ' does eventually achieve the goal, so it must be the case that λ_k achieves the goal. Hence, $\lambda_k \models \mathsf{F}\phi_{\text{goal}}$ applies. Finally, because λ_k is a run starting from $\langle s^k, o^k\rangle$, then $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s^k, o^k\rangle \models \mathsf{EF}\phi_{\text{goal}}$, which is what we aimed to prove. ☐

The above two theorems say that state-strong fair is a *complete* characterization of the type of environment for which strong-cyclic plans are successful. Having now a precise logical characterization of what adequate environments are opens the door for applying (reactive) synthesis techniques to solve FOND problems. That is the topic of the next section.

4. FOND Planning via Reactive Synthesis

The characterization (Theorems 1 and 2) in LTL of the fairness assumption (Definition 6) behind strong-cyclic plans allows one to express the FOND planning problem as a reactive synthesis one (see Section 2.3). Indeed, following (Bloem et al., 2012, Thm. 1) and given a FOND planning problem, we first take the planning propositions as the non-controllable variables (i.e., $\mathcal{X} = P$) and

$\mathcal{Y} = O$ and the operators as the controllable ones. Then, if ϕ_{goal} is the goal of the planning task, we perform LTL synthesis for the following specification formula:

$$\varphi_{\mathcal{P}} \stackrel{\text{def}}{=} \varphi_{\mathcal{P}}^{\mathit{eff}} \wedge (\varphi_{\mathcal{P}}^{\mathit{pre}} \rightarrow [\gamma_{\mathcal{P}}^{\mathit{sfair}} \rightarrow \mathsf{F}\phi_{\text{goal}}]), \tag{2}$$

where, roughly speaking, $\varphi_{\mathcal{P}}^{\mathit{eff}}$ encodes the dynamics of the planning domain (i.e., the effects of actions) and $\varphi_{\mathcal{P}}^{\mathit{pre}}$ encodes the preconditions of operators. Intuitively, specification $\varphi_{\mathcal{P}}$ asks for a policy function such "for every possible generated run (under that policy) that adheres to the effects of operators ($\varphi_{\mathcal{P}}^{\mathit{eff}}$), if every controllable operator prescribed by the policy is legally executed along the run, then provided that the run shows fairness on the operators' effects ($\gamma_{\mathcal{P}}^{\mathit{sfair}}$) the goal will eventually be true (in the run)." Basically, $\varphi_{\mathcal{P}}^{\mathit{eff}}$ and $\varphi_{\mathcal{P}}^{\mathit{pre}}$ encode together the structure $\mathcal{K}_{\mathcal{P}}$ and the initial state of the planning problem by means of safety formulas; (see Bloem et al., 2012).

Notably, synthesis of formula $\varphi_{\mathcal{P}}$ above provides a fully *declarative* approach to FOND planning. There are however significant practical problems with this specification. The fact is that controller synthesis for general LTL specifications is 2EXPTIME-complete and known automata-based techniques have resisted practical implementations due to automata complementation (Kupferman, Piterman, & Vardi, 2006; Pnueli & Rosner, 1989b). This probably explains why the application of controller synthesis as a mean for solving planning problems has been discouraged. Nonetheless, recent advances in controller synthesis (e.g., Bloem et al., 2012; D'Ippolito et al., 2011; Piterman et al., 2006) have shown that restricting the form of the specification system goal allows for more effective solutions *and* implementable systems. Unfortunately, though, the above formula $\varphi_{\mathcal{P}}$ does not fall into known efficient LTL fragments and, what is more, the fairness specification $\gamma_{\mathcal{P}}^{\mathit{sfair}}$ is already exponential in size w.r.t. the planning domain! Consequently, performing LTL synthesis on $\varphi_{\mathcal{P}}$ above would not yield an optimal technique w.r.t. computation complexity (recall that FOND planning is EXPTIME; see Rintanen, 2004).

So, in what follows, we show how to actually solve FOND planning problems via reactive synthesis by suitably encoding the fairness assumption without an increase in complexity.

4.1 Solving FOND Planning via Controller Synthesis in Büchi Games

We reduce here the problem of checking for a strong-cyclic plan to checking for existence of a winning strategy in a particular, computationally amenable, type of two-player game. Further, we show that an actual strong-cyclic plan can be extracted from such a strategy. Notably, the proposed reduction (efficiently) compiles away the state-strong fairness condition, thus not requiring fair realizability/synthesis (Vardi, 1995). The result is a sound and complete *discrete controller synthesis-based* approach to FOND planning that is optimal w.r.t. computational complexity.[9]

More concretely, the game we shall use is that with a *Büchi winning condition*, which, informally, states that "something (good) happens infinitely often." Technically, given a game graph with state space S (c.f. Section 2.3), a Büchi winning condition is defined in terms of a subset $\mathcal{F}$ of S, with the intended meaning that winning plays ought to visit some state in $\mathcal{F}$ infinitely many times. Given a play ρ, we use $\mathit{inf}(\rho)$ to denote the set of states that occur infinitely often in play ρ

Definition 7 (2-player Büchi Games). Given a game graph $\mathcal{G} = \langle (S, E, s_0), (S_\Box, S_\Diamond)\rangle$ and Büchi acceptance condition $\mathcal{F} \subseteq S$, the corresponding ***Büchi game*** is defined as $\mathcal{G}(\mathcal{W}_{\mathcal{F}})$ where $\mathcal{W}_{\mathcal{F}} = \{\rho \mid \mathit{inf}(\rho) \cap \mathcal{F} \neq \emptyset\}$. ■

9. Analogous results can be obtained via synthesis of policies in probabilistic models with probability 1 goal success (Forejt et al., 2011); our work keeps the formalism completely non-probabilistic.

It turns out that every 2-player game with Büchi winning condition for player $\Box$ (2-player Büchi game) is in fact determined (Martin, 1975). Hence, the set of states S in game graph $\mathcal{G}$ is partitioned in those from which $\Box$ has a sure winning strategy ($\mathcal{W}_\Box$) and those states from which $\Diamond$ has a sure winning strategy ($\mathcal{W}_\Diamond$). Also, pure memoryless strategies are enough for 2-player Büchi games: if there is a winning strategy there is a pure memoryless one that is winning.

Let us now provide the main definition of this section, which basically specifies a distinguished 2-player Büchi game for a given FOND planning problem.

Definition 8 (FOND to Büchi Control Problem). Let $\mathcal{P} = \langle P, O, s_I, \phi_{\text{goal}}\rangle$ be a FOND planning problem. The corresponding ***Büchi Control Problem (or Game)*** is a 2-player Büchi game $\mathcal{G}_\mathcal{P}(\mathcal{W}_\mathcal{F})$ with game graph $\mathcal{G}_\mathcal{P} = \langle (S, E, s_0), (S_\Box, S_\Diamond)\rangle$ and Büchi acceptance condition $\mathcal{F}$ defined as follows:

- $s_0 = \langle s_I, \#, \Box\rangle$ as the initial state of the game.
- $S = (2^P \times (O \cup \{\#\}) \times \{\Box, \Diamond\}) \cup \{\langle s^*, o\rangle \mid \ s \in 2^P, o \in O, * \in \{\Box, \Diamond\}\}$ as the set of states of the game.
- $S_\Diamond = \{\langle s^\Diamond, o\rangle \in S\} \cup \{\langle s, o_\#, \Diamond\rangle \in S \mid o_\# \in O \cup \{\#\}\}$ as the set of $\Diamond$-turn states.
- $S_\Box = \{\langle s^\Box, o\rangle \in S\} \cup \{\langle s, o_\#, \Box\rangle \in S \mid \ o_\# \in O \cup \{\#\}\}$ as the set of $\Box$-turn states.
- The transition relation E is defined as follows (recall here that Eff_o is the set of all possible effects of operator O, more than one for non-deterministic operators):

$$\begin{aligned} E \ = \ & \{(\langle s, o_\#, \Box\rangle, \langle s, o, \Diamond\rangle) \mid \ o_\# \in O \cup \{\#\}, o \in O, s \models \mathit{Pre}_o\} \cup \\ & \{(\langle s, o, \Diamond\rangle, \langle s^*, o\rangle) \mid \ o \in O, * \in \{\Box, \Diamond\}\} \cup \\ & \{(\langle s^*, o\rangle, \langle s', o, \Box\rangle) \mid \ o \in O, * \in \{\Box, \Diamond\}, s' \in \mathit{next}(o, s)\} \cup \\ & \{(\langle s, o_\#, \Box\rangle, \langle s, \#, \Box\rangle) \mid \ o_\# \in O \cup \{\#\}, s \models \phi_{\text{goal}}\}. \end{aligned}$$

- $\mathcal{F} = \mathcal{F}_\Box \cup \mathcal{F}_\Diamond$ as the Büchi acceptance condition where $\mathcal{F}_\Box = \{\langle s, o_\#, \Box\rangle \mid s \models \phi_{\text{goal}}, o_\# \in O \cup \{\#\}\}$ and $\mathcal{F}_\Diamond = \{\langle s^\Diamond, o\rangle \in S\}$. ■

Figure 1 depicts a part of the reduction for one step operator. From any $\Box$-state $\langle s, *, \Box\rangle$, there is a transition to $\Diamond$-state $\langle s, o, \Diamond\rangle$ for every operator o that is executable at state s (i.e., $s \models \mathit{Pre}_o$). From such $\Diamond$-state the game can progress to either $\Diamond$-state $\langle s^\Diamond, o\rangle$ or $\Box$-state $\langle s^\Box, o\rangle$, from where one of the possible effects of operator o will be applied, yielding corresponding transitions to states $\langle s'_i, o, \Box\rangle$ such that s'_i is the successor state of applying o's possible effect $e_i \in \mathit{Eff}_o$ in state s. We note that game state $\langle s_I, \#, \Box\rangle$ is the initial state, with $\#$ being a "dummy" fresh symbol. The winning objective for player $\Box$ in $\mathcal{G}$ is defined as all $\Box$-states $\langle s, *, \Box\rangle$ in which the goal holds true in its planning state s (i.e., $s \models \phi_{\text{goal}}$) together with all $\Diamond$-states of the form $\langle s^\Diamond, *\rangle$. The latter will play a key role in the ability to capture a FOND planning task, and in particular, its "fairness" assumption (see below).

In the construction above, informally, a one-step operator execution is modeled as a 3-step process in the game: selection of the operator to be executed, selection of who decides its effects, and realization of one of the possible operator effects. More concretely, the three-step process for an operator execution goes as follows:

Step 1: $\Box$'s turn. At state $\langle s, *, \Box\rangle$ in $\mathcal{G}_\mathcal{P}$, player $\Box$ chooses a successor state $\langle s, o, \Diamond\rangle$ for some executable operator $o \in O$ in s (i.e., one for which $s \models \mathit{Pre}_o$ holds).

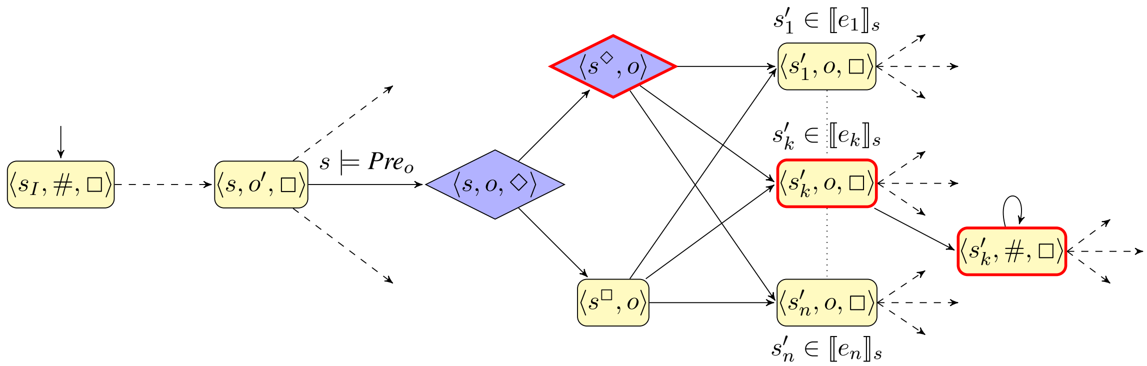


Figure 1: Example of a Reduction to a Buchi Control Game for the case when operator o is performed in state s. Rectangular nodes are $\square$-moves while diamond nodes are $\diamond$-moves. Effects $e_1, \ldots, e_n$, for some $n \geq 1$, are all the possible effects of operator o and s'_k is the only state shown where the goal holds true (i.e., $s'_k \models \phi_{\text{goal}}$). Red thicker borders denote Büchi goal states.

Step 2: $\diamond$'s turn. At state $\langle s, o, \diamond\rangle$ in $\mathcal{G}_\mathcal{P}$, player $\diamond$ chooses a successor between 2 possibilities: $\langle s^\square, o\rangle$ or $\langle s^\diamond, o\rangle$. Basically, the decision that $\diamond$ makes at this step is which player will next decide the effect of the operator, her or player $\square$.

Step 3: $\square$'s turn or $\diamond$'s turn. In states $\langle s^\square, o\rangle$ and $\langle s^\diamond, o\rangle$ players $\square$ and $\diamond$, respectively, choose a successor of the form $\langle s', o, \square\rangle$ such that s' is a possible successor state of state s in planning problem $\mathcal{P}$, that is, one for which there exists an effect $e \in \mathit{Eff}_o$ such that $s' \in [\![e]\!]_s$.

The three-step game $\mathcal{G}_\mathcal{P}(\mathcal{W}_\mathcal{F})$ abstracts away the non-determinism of actions while balancing the power of each player in the game. Possibly the most interesting step in the reduction is the second, in which $\diamond$ decides which player will select the actual operator's effect that should ensue, among the ones possible. Importantly, since $\langle s^\diamond, o\rangle$ is part of the acceptance Büchi set $\mathcal{F}$, the objective for player $\square$ is satisfied—$\square$ "wins"—if game state $\langle s^\diamond, o\rangle$ is visited infinitely often. So, player $\diamond$ pays the price of choosing the planning successor state, when executing operator o in planning state s, by adding a goal state to the play. Put differently, player $\square$ wins, if player $\diamond$ (chooses to) select the operator's effect, in a state, infinitely often. As a consequence, for $\diamond$ to have a chance to win, it cannot decide an operator's effect in a state forever: *eventually it has to yield control to player $\square$ for the selection of the operator's effect*. This means that for $\diamond$ to win, it needs to bring the game—through a *finite* number of effect choices—to a state where $\square$ cannot win anymore even if it is given control of effect selection.

Example 2. In Figure 2 the reduction to the BCP for the planning problem $\mathcal{P}_{coin}$ of Example 1 is shown. It depicts how the three-step game abstracts away the non-determinism of certain actions (like FLIP$_1$ and FLIP$_2$) by balancing the power of each player.

For instance, if player $\diamond$ would be so malicious to synchronize failures ("mimicking" the *unfair* run $\lambda = w_0(w_1w_2w_4w_1w_3w_5)^\omega$), the resulting trace would indeed be winning for $\square$ in the game.■

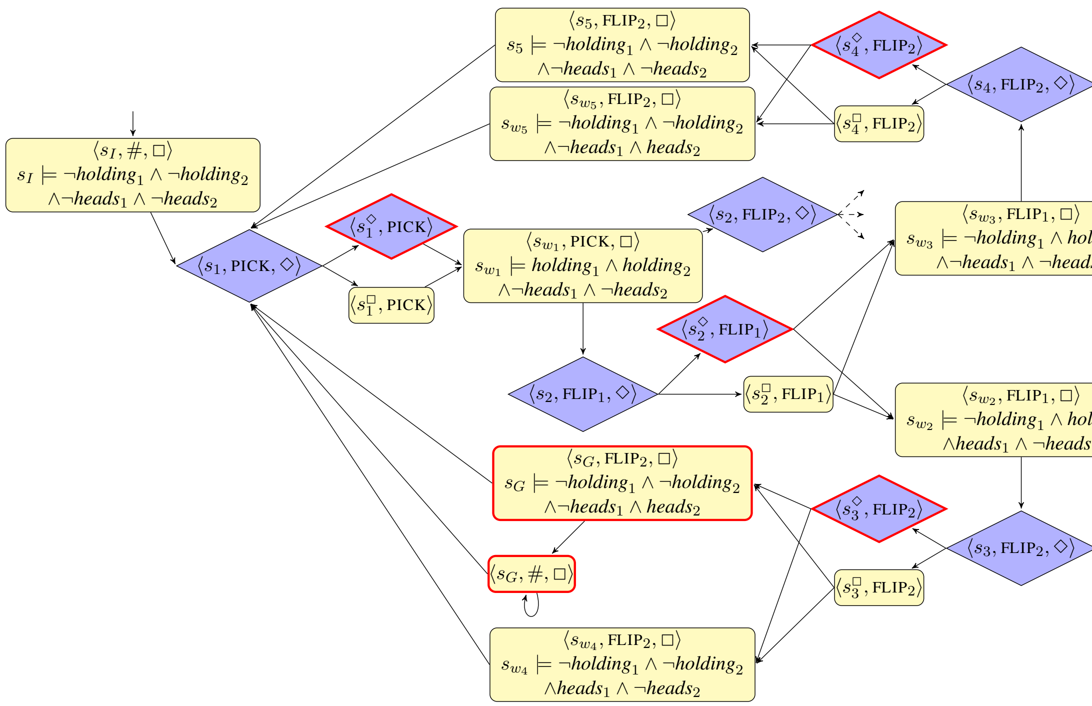


Figure 2: Büchi Control Game $\mathcal{G}_{\mathcal{P}_{coin}}(\mathcal{W}_{\mathcal{F}})$ for the planning problem $\mathcal{P}_{coin}$ from Example 1. States s_{w_k} in the game represent the first component of those w_k belonging to $\mathcal{K}^{\pi}_{\mathcal{P}_{coin}}$ in the "unfair" run λ (refer to Example 1).

As expected, it is possible to extract a planning policy from a player $\Box$'s strategy: operator o is an adequate action to take in planning state s if player $\Box$ happens to choose o in *some* game state having s as the planing state.

Definition 9 (Strategies to Policy). Let $\mathcal{P} = \langle P, O, s_I, \phi_{\text{goal}}\rangle$ be a FOND planning problem and $\mathcal{G}_{\mathcal{P}}(\mathcal{W}_{\mathcal{F}})$ the BCP instance obtained from $\mathcal{P}$ as per Definition 8 above. Let $\sigma_\Box$ be a strategy for player $\Box$ in game $\mathcal{G}_{\mathcal{P}}$. Then, the associated FOND policy $\pi_{\sigma_\Box}$ is defined as follows, for all $s \in 2^P$:

$$\pi_{\sigma_\Box}(s) = \{o \mid o \in O, i \in \mathbb{N}, \sigma_\Diamond \in \Sigma_\Diamond, \rho^{\sigma_\Box,\sigma_\Diamond}[i] = \langle s, *, \Box\rangle, \sigma_\Box(\langle s, *, \Box\rangle) = \langle s, o, \Diamond\rangle\}. \qquad \blacksquare$$

Observe that the associated policy $\pi_{\sigma_\Box}$ may be non-deterministic, as $\sigma_\Box$ may prescribe different operators for two different game states of $\mathcal{G}_P$ sharing the same first component.

The following result makes the connection between runs in $\mathcal{P}$-structure $\mathcal{K}_{\mathcal{P}}$ (Definition 4) and the game graph $\mathcal{G}_{\mathcal{P}}$ with respect to a planning task $\mathcal{P}$.

Lemma 1. Let $\mathcal{P}$ be a FOND planning problems. Then (here $s^0 = s_I$, the initial state of $\mathcal{P}$):

$$\langle s^0, o^0\rangle\langle s^1, o^1\rangle \cdots \langle s^{k-1}, o^{k-1}\rangle\langle s^k, o^k\rangle, \text{ with } k \geq 0, \text{ is a (finite) prefix of a run in } \mathcal{K}_{\mathcal{P}}$$
iff

$$\langle s_I, \#, \Box\rangle\langle s^0, o^0, \Diamond\rangle\langle x^0, o^0\rangle\langle s^1, o^0, \Box\rangle\langle s^1, o^1, \Diamond\rangle\langle x^1, o^1\rangle\langle s^2, o^1, \Box\rangle\langle s^2, o^2, \Diamond\rangle \cdots$$
$$\cdots \langle s^{k-1}, o^{k-1}, \Diamond\rangle\langle x^{k-1}, o^{k-1}\rangle\langle s^k, o^{k-1}, \Box\rangle\langle s^k, o^k, \Diamond\rangle, \text{ with } x^i \in \{s^{i^\Box}, s^{i^\Diamond}\} \text{ and } o^i \in O, \text{ for all } i \in \{0, \ldots, k-1\}, \text{ is a (finite) prefix of a play in game graph } \mathcal{G}_\mathcal{P}.$$

Proof. We prove this by induction on the length n of prefix of a run in $\mathcal{K}_\mathcal{P}$:

- If $n = 0$, then we aim to prove that $\langle s^0, o^0\rangle$ is a (finite) prefix of a run in $\mathcal{K}_\mathcal{P}$ iff it is the case that $\langle s_I, \#, \Box\rangle\langle s^0, o^0, \Diamond\rangle$ is a prefix of a play in game graph $\mathcal{G}_\mathcal{P} = \langle(S, E, s_0), (S_\Box, S_\Diamond)\rangle$. This follows trivially from Definition 8 and the fact that $\langle s^0, o^0\rangle$ belongs to the run as:
  - $s_0 = \langle s_I, \#, \Box\rangle$ is the initial state of the game and $s^0 = s_I$ is $\mathcal{P}$'s initial state; and
  - $s^0 \models \mathit{Pre}_{o^0}$ iff $(\langle s_I, \#, \Box\rangle, \langle s^0, o^0, \Diamond\rangle) \in E$.
- Assume that the thesis holds for any $n = k$, with $k \geq 1$ (induction hypothesis), and consider now $n = k + 1$ (which implies that $n \geq 1$).

  ONLY-IF. First, suppose that $\chi = \langle s^0, o^0\rangle\langle s^1, o^1\rangle \cdots \langle s^k, o^k\rangle\langle s^{k+1}, o^{k+1}\rangle$ is a (finite) prefix of a run in $\mathcal{K}_\mathcal{P} = \langle W, R, P\rangle$. By the induction hypothesis we have that

  $$\rho_k = \langle s_I, \#, \Box\rangle\langle s^0, o^0, \Diamond\rangle \cdots\cdots \langle s^{k-1}, o^{k-1}, \Diamond\rangle\langle x^{k-1}, o^{k-1}\rangle\langle s^k, o^{k-1}, \Box\rangle\langle s^k, o^k, \Diamond\rangle, \text{ with } x^i \in \{s^{i^\Box}, s^{i^\Diamond}\} \text{ and } o^i \in O, \text{ for all } i \in \{0, \ldots, k-1\}, \text{ is a (finite) prefix of a play in game graph } \mathcal{G}_\mathcal{P}.$$

  Now, it follows from χ that $R(\langle s^k, o^k\rangle\langle s^{k+1}, o^{k+1}\rangle)$ in $\mathcal{K}_\mathcal{P}$. Then, using Definitions 3 and 4 for $\mathcal{K}_\mathcal{P}$ and the notion of policies, we know that $s^k \models \mathit{Pre}_{o^k}$ and $s^{k+1} \models \mathit{Pre}_{o^{k+1}}$ (as $o^k \in \pi(\hat{s^k})$ and $o^{k+1} \in \pi(\hat{s^{k+1}})$), and $s^{k+1} \in \mathit{next}(o^k, s^k)$.

  With this, we conclude the existence of the following three pairs in $\mathcal{G}_\mathcal{P}$'s relation E as per Definition 8:
  - Because $s^k \models \mathit{Pre}_{o^k}$, it follows that $(\langle s^k, o^k, \Diamond\rangle, \langle x^k, o^k\rangle) \in E$, with $x^k \in \{s^{k^\Box}, s^{k^\Diamond}\}$.
  - Because $s^{k+1} \in \mathit{next}(o^k, s^k)$, it follows that $(\langle x^k, o^k\rangle, \langle s^{k+1}, o^k, \Box\rangle) \in E$.
  - Finally, $(\langle s^{k+1}, o^k, \Box\rangle, \langle s^{k+1}, o^{k+1}, \Diamond\rangle) \in E$ applies directly.

  Putting all together, we conclude that $\rho_k\langle x^k, o^k\rangle\langle s^{k+1}, o^k, \Box\rangle\langle s^{k+1}, o^{k+1}, \Diamond\rangle)$ is also a (finite) prefix of a play in game graph $\mathcal{G}_\mathcal{P}$.

(IF.) Suppose that

$$\rho_{k+1} = \langle s_I, \#, \Box\rangle\langle s^0, o^0, \Diamond\rangle \cdots\cdots \langle s^k, o^k, \Diamond\rangle\langle x^k, o^k\rangle\langle s^{k+1}, o^k, \Box\rangle\langle s^{k+1}, o^{k+1}, \Diamond\rangle,$$

with $x^i \in \{s^{i^\Box}, s^{i^\Diamond}\}$ and $o^i \in O$, is a (finite) prefix of a play in game graph $\mathcal{G}_\mathcal{P}$.

By induction hypothesis, $\chi_k = \langle s^0, o^0\rangle\langle s^1, o^1\rangle \cdots \langle s^k, o^k\rangle\langle s^k, o^k\rangle$ is a (finite) prefix of a run in $\mathcal{K}_\mathcal{P} = \langle W, R, P\rangle$. We are to prove that $R(\langle s^k, o^k\rangle, \langle s^{k+1}, o^{k+1}\rangle)$.

Due to ρ_{k+1}, we have $(\langle s^{k+1}, o^k, \Box\rangle\langle s^{k+1}, o^{k+1}, \Diamond\rangle) \in E$ and thus $s^{k+1} \models \mathit{Pre}_{o^{k+1}}$. By definition of the universal policy $\hat{\pi}$, it follows that $o^{k+1} \in \hat{\pi}(s^{k+1})$. In addition, from ρ_{k+1} again, we know that $(\langle x^k, o^k\rangle, \langle s^{k+1}, o^k, \Box\rangle) \in E$, which implies that $s^{k+1} \in \mathit{next}(o^k, s^k)$. Putting both together, and following Definition 4, it follows that $R(\langle s^k, o^k\rangle, \langle s^{k+1}, o^{k+1}\rangle)$ is true and that $\rho_{k+1}\langle s^{k+1}, o^{k+1}\rangle$ is a (finite) prefix of a run in $\mathcal{K}_\mathcal{P}$. $\Box$

The next main result shows that the induced FOND policy obtained from a solution to the corresponding Büchi Control Problem (i.e., a winning strategy for player $\Box$) is indeed a strong-cyclic solution for the planning task of concern.

Theorem 3 (Correctness). Let $\mathcal{P}$ be a FOND planning problem and $\mathcal{G}_{\mathcal{P}}$ its associated Büchi Control Problem as per reduction in Definition 8. Then, $\langle s_I, \#, \Box \rangle \in \mathcal{W}_{\Box}$ (i.e., there exists a sure winning strategy for $\Box$ from the initial state $\langle s_I, \#, \Box \rangle$ in $\mathcal{G}_{\mathcal{P}}(\mathcal{W}_{\mathcal{F}})$) *iff* there exists a strong-cyclic policy π for $\mathcal{P}$.

Proof. **(ONLY-IF)** Let $\sigma_{\Box} \in \Sigma_{\Box}$ be a sure winning strategy for $\Box$ from $\langle s_I, \#, \Box \rangle$ in $\mathcal{G}_{\mathcal{P}}$. Wlog, we assume that $\sigma_{\Box}$ is *uniform*, in that it always makes the same operator choices in the states sharing the same planning state. Formally, for all $\langle s, o_1, \Box \rangle, \langle s, o_2, \Box \rangle \in S_{\Box}$ with $o_1, o_2 \in O$, it is the case that $\sigma_{\Box}(\langle s, o_1, \Box \rangle) = \sigma_{\Box}(\langle s, o_2, \Box \rangle)$.

Let π be its associated FOND policy as per Definition 9. We shall prove next that π is a strong-cyclic solution plan for $\mathcal{P}$, that is, as per Definition 5, we are to prove

$$\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s_I, \pi(s_I) \rangle \models \mathsf{A}(\mathsf{EF}\phi_{\text{goal}} \mathsf{W} \phi_{\text{goal}}).$$

To that end, consider an arbitrary maximal run $\lambda = \langle s^0, \pi(s^0) \rangle \langle s^1, \pi(s^1) \rangle \cdots$ of $\mathcal{K}^{\pi}_{\mathcal{P}}$ from state $s^0 = s_I$, and a state $\langle s^{\ell}, \pi(s^{\ell}) \rangle$ in λ for some $\ell \geq 0$ such that for all $i \leq \ell$, $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s_i, \pi(s_i) \rangle \models \neg\phi_{\text{goal}}$ (i.e., $s_i \models \neg\phi_{\text{goal}}$). Due to the semantics of the weak-until quantifier, we must prove $\mathcal{K}^{\pi}_{\mathcal{P}}, \langle s^{\ell}, \pi(s^{\ell}) \rangle \models \mathsf{EF}\phi_{\text{goal}}$, that is, the goal is reachable from $\langle s^{\ell}, \pi(s^{\ell}) \rangle$ in $(\mathcal{P}, \pi)$-structure $\mathcal{K}^{\pi}_{\mathcal{P}}$.

Next consider the finite prefix run $\lambda_{\ell} = \langle s^0, \pi(s^0) \rangle \cdots \langle s^{\ell}, \pi(s^{\ell}) \rangle$ of λ. Due to Lemma 1, there exists a prefix play in the game graph $\mathcal{G}_{\mathcal{P}}$ "mimicking" run λ_{ℓ} of the form:

$$\begin{aligned}\rho_{\lambda_{\ell}} = \langle s_I, \#, \Box \rangle \langle s^0, \pi(s^0), \Diamond \rangle \langle {s^0}^{\Diamond}, \pi(s^0) \rangle \langle s^1, \pi(s^0), \Box \rangle \langle s^1, \pi(s^1), \Diamond \rangle \cdots \\ \cdots \langle s^{\ell-1}, \pi(s^{\ell-1}), \Diamond \rangle \langle {s^{\ell-1}}^{\Diamond}, \pi(s^{\ell-1}) \rangle \langle s^{\ell}, \pi(s^{\ell-1}), \Box \rangle.\end{aligned}$$

That is, $\rho_{\lambda_{\ell}}$ is the play in the game corresponding to finite trace λ_{ℓ} in which each operator effect is decided by player $\Diamond$ (the "opponent").

Next, we extend prefix play $\rho_{\lambda_{\ell}}$ to an infinite play $\rho^{+}_{\lambda_{\ell}}$ compatible with $\sigma_{\Box}$ for $\Box$ and a strategy for $\Diamond$ that lets player $\Box$ decide (from then on) the effect of each operator in the game:

$$\rho^{+}_{\lambda_{\ell}} = \rho_{\lambda_{\ell}} \langle {s^{\ell}}^{\Box}, \pi(s^{\ell}) \rangle \langle q^1, \pi(s^{\ell}), \Box \rangle \langle q^1, o^1, \Diamond \rangle \langle {q^1}^{\Box}, o^1 \rangle \langle q^2, o^1, \Box \rangle \langle q^2, o^2, \Diamond \rangle \langle {q^2}^{\Box}, o^2 \rangle \cdots$$

So, $\rho^{+}_{\lambda_{\ell}}$ is a play in $\mathcal{G}_{\mathcal{P}}$ that is compatible with $\sigma_{\Box}$ for $\Box$ (because of the way π was extracted from $\sigma_{\Box}$ and the uniformity assumption on $\sigma_{\Box}$) and a strategy $\sigma_{\Diamond} \in \Sigma_{\Diamond}$ for player $\Diamond$ that decides the effect for the first ℓ-th operators—play prefix $\rho_{\lambda_{\ell}}$—and then always lets player $\Box$ decide the effect of all subsequent operators. Formally, strategy $\sigma_{\Diamond}$ is such that:

- $\sigma_{\Diamond}(\langle s^i, \pi(s^i), \Diamond \rangle) = \langle {s^i}^{\Diamond}, \pi(s^i) \rangle$ and $\sigma_{\Diamond}(\langle {s^i}^{\Diamond}, \pi(s^i) \rangle) = \langle s^{i+1}, \pi(s^i), \Box \rangle$, for all $i = \{0, \ldots, \ell - 1\}$;
- $\sigma_{\Diamond}(\langle s^{\ell}, \pi(s^{\ell}), \Diamond \rangle) = \langle {s^{\ell}}^{\Box}, \pi(s^{\ell}) \rangle$; and
- $\sigma_{\Diamond}(\langle o^i, o^i, \Diamond \rangle) = \langle {s^i}^{\Box}, o^i \rangle$ and $\sigma_{\Diamond}(\langle {o^i}^{\Box}, o^i \rangle) = \langle o^{i+1}, o^i), \Box \rangle$, for all $i \geq 1$.

Since $\sigma_\Box$ is a winning strategy for player $\Box$ it has to be the case that $\mathit{inf}(\rho^+_{\lambda_\ell}) \cap \mathcal{F} \neq \emptyset$. Moreover, as the selected $\sigma_\Diamond$ selects a state of the form $\langle s^\Diamond, o\rangle$ only a finite number of times (namely, the first ℓ steps), it follows that $\mathit{inf}(\rho^+_{\lambda_\ell}) \cap \mathcal{F}_\Diamond = \emptyset$. Since $\mathcal{F} = \mathcal{F}_\Diamond \cup \mathcal{F}_\Box$, it has to be the case that $\mathit{inf}(\rho^+_{\lambda_\ell}) \cap \mathcal{F}_\Box \neq \emptyset$. This implies that there exists $k \geq 1$ such that game state $\langle q^k, o^k, \Box\rangle$ appears an infinite number of times in play $\rho^+_{\lambda_\ell}$ and $q^k \models \phi_{\text{goal}}$, that is, the play does reach a state where the planning goal holds. Wlog, assume k to be the minimum one, that is, q^k is the first state in play $\rho^+_{\lambda_\ell}$ where the planning goal is true (i.e., $q^i \not\models \phi_{\text{goal}}$, for all $i < k$).

The remaining of the proof involves building back, from play $\rho^+_{\lambda_\ell}$ and via Lemma 1, a run in $\mathcal{K}^\pi_\mathcal{P}$ showing that the goal is reachable from state $\langle s^\ell, \pi(s^\ell)\rangle$. First consider the following prefix of $\rho^+_{\lambda_\ell}$:

$$\begin{aligned}\rho^+_k = \rho_{\lambda_\ell}\langle s^{\ell\Box}, \pi(s^\ell)\rangle\langle q^1, \pi(s^\ell), \Box\rangle\langle q^1, o^1, \Diamond\rangle\langle q^{1\Box}, o^1\rangle\langle q^2, o^1, \Box\rangle\langle q^2, o^2, \Diamond\rangle \cdots \\ \cdots\langle q^{k-1}, o^{k-2}, \Box\rangle\langle q^{k-1}, o^{k-1}, \Diamond\rangle\langle q^{k-1\Box}, o^{k-1}\rangle\langle q^k, o^{k-1}, \Box\rangle.\end{aligned}$$

Because q^k is the first game state where the planning goal ϕ_{goal} holds true, it follows from the structure of the game graph that $o^i \neq \#$ and hence $o^i \in O$, for all $i \in \{1, \ldots, k-1\}$. Moreover, since $\rho^+_{\lambda_\ell}$ is compatible with winning strategy $\sigma_\Box$, from which uniform policy π was extracted, it is the case that $o^i = \pi(q^i)$, for all $i \in \{1, \ldots, k-1\}$, and we can re-write ρ^+_k as follows:

$$\begin{aligned}\rho^+_k = \rho_{\lambda_\ell}\langle s^{\ell\Box}, \pi(s^\ell)\rangle\langle q^1, \pi(s^\ell), \Box\rangle\langle q^1, \pi(q^1), \Diamond\rangle\langle q^{1\Box}, \pi(q^1)\rangle\langle q^2, \pi(q^1), \Box\rangle\langle q^2, \pi(q^2), \Diamond\rangle \cdots \\ \cdots\langle q^{k-1}, \pi(q^{k-2}), \Box\rangle\langle q^{k-1}, \pi(q^{k-1}), \Diamond\rangle\langle q^{k-1\Box}, \pi(q^{k-1})\rangle\langle q^k, \pi(q^{k-1}), \Box\rangle.\end{aligned}$$

Now, due to Lemma 1, there exists a prefix of a maximal run in structure $\mathcal{K}_\mathcal{P}$ of the form:

$$\lambda^+_k = \lambda_\ell\langle q^1, \pi(q^1)\rangle\langle q^2, \pi(q^2)\rangle \cdots \langle q^{k-1}, \pi(q^{k-1})\rangle\langle q^k, o\rangle.$$

Since a prefix of a maximal run, the last state in λ_ℓ is $\langle s^\ell, \pi(s^\ell)\rangle$, and all operators are those prescribed by policy π, there is a run in $\mathcal{K}^\pi_\mathcal{P}$ from state $\langle s^\ell, \pi(s^\ell)\rangle$ of the form:

$$\langle s^\ell, \pi(s^\ell)\rangle\langle q^1, \pi(q^1)\rangle\langle q^2, \pi(q^2)\rangle \cdots \langle q^{k-1}, \pi(q^{k-1})\rangle\langle q^k, o\rangle \cdots$$

Finally, since $q^k \models \phi_{\text{goal}}$, it follows from this run that $\mathcal{K}^\pi_\mathcal{P}, \langle s^\ell, \pi(s^\ell)\rangle \models \mathsf{EF}\phi_{\text{goal}}$ applies.

(IF) Assume then that π is a strong-cyclic solution plan for $\mathcal{P}$ and let us prove that there exists a sure winning strategy $\sigma_\Box$ for player $\Box$. We extract such strategy $\sigma_\Box$ from the strong-cyclic plan π as follows:

$$\sigma_\Box(x) = \begin{cases} \langle s, \pi(s), \Diamond\rangle & \text{for} \quad x = \langle s, o, \Box\rangle, o \in O, s \models \neg\phi_{\text{goal}} \\ \langle s, \#, \Box\rangle & \text{for} \quad x = \langle s, o_\#, \Box\rangle, o_\# \in O \cup \{\#\}, s \models \phi_{\text{goal}} \\ \langle s', \pi(s), \Box\rangle & \text{for} \quad x = \langle s^\Box, \pi(s)\rangle, s' = \mathit{nextint}(s, o, \phi_{\text{goal}}) \end{cases}$$

where $\mathit{nextint}(s, o, \phi_{\text{goal}})$ is a function returning a successor state s' of state s after execution of operator o which is "closest" to goal ϕ_{goal} as per policy π (that is, a state s' among set $\mathit{next}(s, o)$ with the shortest path to a goal state in structure $\mathcal{K}^\pi_\mathcal{P}$). It is important to point out that, while function strategy $\sigma_\Box$ will be, in general, partial (e.g., it is undefined for game states $\langle s^\Box, o\rangle$ with $o \neq \pi(s)$), it will indeed be defined in all game states reachable from the initial one, the relevant ones. In particular, $\sigma_\Box$ is defined for every game state $\langle s^\Box, \pi(s)\rangle$ if s is reachable from s_I in structure $\mathcal{K}^\pi_\mathcal{P}$, as π is strong-cyclic and hence closed.

Next, we shall prove that the strategy $\sigma_\Box$ is indeed a sure winning strategy for player $\Box$. Intuitively, if player $\Diamond$ eventually allows $\Box$ to select the effects of operators, $\Box$ will select those leading closer to the goal (a per strong-cyclic policy π). So, consider any arbitrary strategy $\sigma_\Diamond$ for $\Diamond$ and a play ρ compatible with both with $\sigma_\Box$ and $\sigma_\Diamond$

$$\rho = \langle s_I, \#, \Box\rangle\langle s^0, \pi(s^0), \Diamond\rangle\langle \omega^0, \pi(s^0)\rangle\langle s^1, \pi(s^0), \Box\rangle\langle s^1, \pi(s^1), \Diamond\rangle \\ \langle \omega^1, \pi(s^1)\rangle\langle s^2, \pi(s^1), \Box\rangle\langle s^2, \pi(s^2), \Diamond\rangle\langle \omega^2, \pi(s^2)\rangle\langle s^3, \pi(s^1), \Box\rangle \cdots ,$$

where $\omega^i \in \{s^{i^\Diamond}, s^{i^\Box}\}$, for all $i \geq 0$. We want to show that ρ is a winning play for $\Box$, that is, $\mathit{inf}(\rho) \cap \mathcal{F} \neq \emptyset$. To that end, suppose that $\mathit{inf}(\rho) \cap \mathcal{F}_\Diamond = \emptyset$ (otherwise ρ is winning already for $\Box$ as player $\Diamond$ is not being fair in selecting the effect of actions), and let us prove that $\mathit{inf}(\rho) \cap \mathcal{F}_\Box \neq \emptyset$. We point out that this is exactly where the fact that a Büchi game, rather than a reachability one, is important so as to capture fair/unfair runs. So, technically, we are to show that there exists a game state $\langle s, o_\#, \Box\rangle$, with $s \models \phi_{\text{goal}}$, that is mentioned an infinite number of times in play ρ.

From $\mathit{inf}(\rho) \cap \mathcal{F}_\Diamond = \emptyset$, together with the fact that the set of game states is finite, it follows that there exists $\ell \geq 0$ such that $\omega^i = s^{i^\Box}$, for all $i \geq \ell$. That is, from step ℓ onwards, player $\Diamond$ always allows player $\Box$ to pick operators' effects, that is, $\sigma_\Diamond(\langle s^i, \pi(s^i), \Diamond\rangle) = \langle s^{i^\Box}, \pi(s^i)\rangle$, for all $i \geq \ell$.

Now, observe that from step ℓ in ρ, strategy $\sigma_\Box$, by definition, selects the effect of each executed operator that will bring the goal in the shortest path. Remember also that $\sigma_\Box$ follows strong-cyclic solution π, and hence $\sigma_\Box$ is always able to select an operator leading eventually to the goal. So, suppose that the shortest path to a goal state from planning state s^ℓ is $k \geq 0$. Then, play ρ above is of the form:

$$\rho = \langle s_I, \#, \Box\rangle \cdots \langle s^\ell, \pi(s^{\ell-1}), \Box\rangle\langle s^\ell, \pi(s^\ell), \Diamond\rangle\langle s^{\ell^\Box}, \pi(s^\ell)\rangle\langle s^{\ell+1}, \pi(s^\ell), \Box\rangle \cdots \\ \cdots \langle s^{\ell+k-1}, \pi(s^{\ell+k-1}), \Diamond\rangle\langle s^{\ell+k-1^\Box}, \pi(s^{\ell+k-1})\rangle\langle s^{\ell+k}, \pi(s^{\ell+k-1}), \Box\rangle\theta_1\theta_2 \cdots ,$$

such that $s^{\ell+k} \models \phi_{\text{goal}}$, as each operator executed yields an effect following the shortest path to the goal in $\mathcal{K}^\pi_\mathcal{P}$. Then, due to the way that $\sigma_\Box$ was defined, it follows that $\theta_i = \langle s^{\ell+k}, \#, \Box\rangle$, for all $i \geq 1$—that is, a goal game state has been reached, so $\sigma_\Box$ forces the game to remain in $\langle s^{\ell+k}, \#, \Box\rangle$ forever. This means that $\langle s^{\ell+k}, \#, \Box\rangle \in \mathit{inf}(\rho) \cap \mathcal{F}_\Box$, play ρ is a winning play for $\Box$, and, since ρ was arbitrary chose, $\sigma_\Box$ is a sure winning strategy for player $\Box$ (hence, there could be no winning strategy for player $\Diamond$ and the thesis follows). $\Box$

Correctness states that the problem of finding a sure winning strategy for player $\Box$ in the corresponding game $\mathcal{G}$ is equivalent to the problem of finding a strong-cyclic policy π for $\mathcal{P}$. Definition 9 states how to construct such a policy based on the sure winning strategy for player $\Box$ that can be found in polynomial-time. Thus, the complexity of solving the problem of finding a strong-cyclic policy π for $\mathcal{P}$ is in PTIME.

In turn, if there is no strong-cyclic solution for the planning task, then player $\Diamond$ ought to have a winning strategy, and vice-versa. This follows almost directly from Theorem 3 and the fact that the game $\mathcal{G}_\mathcal{P}(\mathcal{W}_\mathcal{F})$ is determined.

Corollary 1 (Completeness). Let $\mathcal{P}$ be a FOND planning problem and $\mathcal{G}_\mathcal{P}$ its associated Büchi Control Problem as per reduction in Definition 8. Then, there is no strong-cyclic policy π for $\mathcal{P}$ *iff* $\Diamond$ has a sure winning strategy from the initial state $\langle s_I, \#, \Box\rangle$ in $\mathcal{G}_\mathcal{P}$.

Proof. Suppose that there is no strong-cyclic policy for $\mathcal{P}$. Then, due to Theorem 3, there is no sure winning strategy for $\Box$ from the initial state $\langle s_I, \#, \Box \rangle$. Since $\mathcal{G}_{\mathcal{P}}(\mathcal{W}_{\mathcal{F}})$ is determined—every Büchi game is—it follows that there exists a sure winning strategy for player $\Diamond$. The second part follows in analogous way: if there is no sure winning strategy for $\Box$ from game state $\langle s_I, \#, \Box \rangle$, then there ought to be a sure winning strategy for player $\Diamond$ (determinacy of $\mathcal{G}_{\mathcal{P}}(\mathcal{W}_{\mathcal{F}})$) and hence there is no strong-cyclic solution for $\mathcal{P}$ due to Theorem 3. $\Box$

The results above demonstrate that FOND planning can be solved via reactive synthesis. In fact, the two-player game interpretation of controller synthesis allows to compactly and explicitly encode planning problems as Büchi games. Clearly, the specific Büchi game $\mathcal{G}_{\mathcal{P}}$ we obtain is exponential on the succinct representation $\mathcal{P}$ of the planning problem (Definition 4). Thus, since solving Büchi games is known to require polynomial time on the size of the game (Maler, Pnueli, & Sifakis, 1995), we obtain the following complexity lower-bound for our realizability/synthesis based approach to FOND planning.

Theorem 4 (Complexity). Let $\mathcal{P} = \langle P, O, s_I, \phi_{\text{goal}} \rangle$ be a FOND planning problem and $\mathcal{G}_{\mathcal{P}}$ its associated Büchi Control Problem as per Definition 8. Deciding whether $\mathcal{G}_{\mathcal{P}}$ is winning and computing a winning strategy for $\Box$ can be done in $\mathrm{O}((4 \times |2^P| \times |O|)^2)$.

Proof. Deciding whether a winning strategy for two-player games with deterministic Büchi conditions exist, and computing one, is known to be quadratic in the size of the game's state space (Maler et al., 1995), in our case, set S of $\mathcal{G}_{\mathcal{P}}$. Since S is $4 \times |O|$ times *larger* than $\mathcal{K}_{\mathcal{P}}$'s state space, we have that $|S| \in \mathrm{O}(4 \times (|2^P| \times |O|))$, and the thesis follows. $\Box$

This is, up to our knowledge, the first realizability/synthesis approach to FOND planning that is optimal w.r.t. computational complexity, in that it solves the problem in quadratic time w.r.t. the size of the input (which is exponential) and thus remains EXPTIME-complete (instead of 2EXPTIME-complete via general LTL controller synthesis). Now, we remind the reader that one can encode a transition system such as $\mathcal{G}_{\mathcal{P}}$ as LTL *safety*-based formulas (Bloem et al., 2012; Clarke, Grumberg, & Peled, 1999), as done in Equation 2. In addition, one can do that succinctly, by directly translating the factored planning description $\mathcal{P}$ into LTL safety formulas. Similarly, the Büchi winning condition of game $\mathcal{G}_{\mathcal{P}}(\mathcal{W}_{\mathcal{F}})$ can be specified succinctly in a polynomially large *weak fairness* LTL formula of the form $\mathsf{GF}\phi_{\mathcal{F}}$. Putting it all together, we get the following summarizing result:

Corollary 2. Let $\mathcal{P}$ be a FOND planning problem with goal ϕ_{goal}. The following statements are equivalent:

1. There exists a strong-cyclic policy solution for $\mathcal{P}$.
2. LTL formula $\varphi^{\mathit{eff}}_{\mathcal{P}} \wedge (\varphi^{\mathit{pre}}_{\mathcal{P}} \rightarrow [\gamma^{\mathit{sfair}}_{\mathcal{P}} \rightarrow \mathsf{F}\phi_{\text{goal}}])$, as per Equation 2, is realizable.
3. There is a sure winning strategy for $\Box$ from the initial state $\langle s_I, \#, \Box \rangle$ in $\mathcal{G}_{\mathcal{P}}$.
4. LTL formula $\varphi^{\Diamond}_{\mathcal{G}_{\mathcal{P}}} \wedge (\varphi^{\Box}_{\mathcal{G}_{\mathcal{P}}} \rightarrow \mathsf{GF}\phi_{\mathcal{F}})$ is realizable, where $\varphi^{\Diamond}_{\mathcal{G}_{\mathcal{P}}}$ and $\varphi^{\Box}_{\mathcal{G}_{\mathcal{P}}}$ are factorized safety formulas encoding game graph $\mathcal{G}_{\mathcal{P}}$ and $\phi_{\mathcal{F}}$ is the factorized formula encoding the Büchi winning condition.

Proof. Equivalence between (1) and (2) follows from Theorems 1 and 2, together with Equation (2). Equivalence among (1), (3), and (4) follows from Theorem 3, and the fact that one can encode Büchi games with safety and weak fairness formulas, as explained above. $\Box$

We close by observing that the reduction developed leverages on the fact that the strong fairness assumption present in FOND planning is of restricted (local) form (see discussion in Section 5).

5. Related Work

The foundations of non-deterministic planning under full observability has been first given by Daniele et al. (2000) and then slightly revised by Pistore and Traverso (2001) to account for arbitrary behavior after the goal has been achieved. There, the notion of strong-cyclic plans as an adequate solution concept for FOND planning was introduced and characterized logically in CTL. The characterization basically states that strong-cyclic plans are those that yield a "live" and "closed" execution graph, that is, one in which the goal is reachable in every state along any possible execution. In other words, regardless of the effects ensuing, the goal is never "lost." Interestingly, such characterization does not refer to the fairness assumption on the environment commonly used in the literature (e.g., Cimatti et al., 2003; Geffner & Bonet, 2013; Kuter et al., 2008; Muise, Belle, & McIlraith, 2014; Ramirez & Sardina, 2014) to frame strong-cyclic policies as those such that *"provided the environment is fair w.r.t. operators' effects, the goal will be achieved."* While most works discuss this at an informal level, Geffner and Bonet (2013) define strong-cyclic policies in those terms formally, albeit in semantic terms by defining what "fair" runs are at the meta-level. Our work provides a logical characterization in CTL of this common understanding of strong-cyclic policies, thus *complementing* the original characterization of Daniele et al. and Pistore and Traverso.

As a result of the growing interest beyond classical planning, there are today notable FOND planning techniques and technologies, such as PRP (Muise, McIlraith, & Beck, 2012) (arguably, the state-of-the-art today), NDP (Kuter et al., 2008), FIP (Fu et al., 2011), MBP (Cimatti et al., 2003), and GAMER (Kissmann & Edelkamp, 2009). The first three are built on top of classical planners, whereas the last two perform (nested) fix-point reasoning via model-checking type techniques. By implementing different clever techniques, they all seek structures that are closed and "live," thus aiming to meet Daniele et al.'s logical specification of strong-cyclic policies. Our account in Section 3, instead, provides the formal justification for applying reactive synthesis *directly against a declarative specification* of strong-cyclic solutions, based on the achievement of the goal under fair environments. While in Section 4 we demonstrated that the FOND complexity can be matched, by a succinct encoding of the effect fairness assumption, we recognize that further work is required to understand which approaches and techniques are more appropriate for different domains and contexts (e.g., whether we are after one solution plan or "universal" type of plans), as well as to integrate diverse approaches into "hybrid" versions (e.g., Ramirez & Sardina, 2014).

When seen as a controller synthesis problem, it follows from Equation 2 that FOND planning amounts to synthesis under fairness assumptions (Vardi, 1995). In particular, the fairness assumption used is a strong fairness condition of the form "executing an action (in a state) infinitely often, yields, immediately, all its effects infinitely often." While feasible, synthesis under fairness assumptions is computationally very demanding (Vardi, 1995 has proved it to be complete for 2EXPTIME). However, the specific fairness condition needed for capturing FOND planning is restricted to a (local) *reactivity condition* of the form $(\mathsf{GF}p \rightarrow \mathsf{GF}q)$ where q is an immediate consequence of p. Our translation (Definition 8) exploits this and transfers the reactivity required in each state in FOND to the three-step Büchi game. More precisely, the three-step interaction of the players "simulates" the (limited) strong fairness condition. We note that our game is inspired by the reduction from stochastic to non-stochastic games with parity conditions developed by Chatterjee et al. (2004).

Given the above, one may wonder whether one could resort to Generalized Reactivity(1), commonly1 referred as GR(1), synthesis for FOND planning. GR(1) is a well-known fragment of LTL that is both highly expressive and significantly better behaved computationally for realizability and synthesis than full LTL (Bloem et al., 2012; Piterman et al., 2006). A GR(1) specification has the form $(\mathsf{GF}\phi_1 \wedge \cdots \wedge \mathsf{GF}\phi_n) \rightarrow (\mathsf{GF}\psi_1 \wedge \cdots \wedge \mathsf{GF}\psi_m)$, thus allowing to specify fairly general *assumption-requirement* specifications. However, one can see that such type of formula will account *directly* for only one state-strong fair constraint, as per Definition 6. To account for all required constraints in a domain, we would need a GR(n) specification. The work of De Giacomo, Patrizi, and Sardina (2010) tries to achieve a GR(1) formula via a natural encoding of strong fairness into weak fairness, as proposed by Kesten, Piterman, and Pnueli (2005). Unfortunately, such encoding only works in the context of model checking, but not synthesis. Said so, GR(1) specifications (and synthesis) is *strictly* more expressive than Büchi conditions, and it is easy to see that our above encoding could be represented as a GR(1) specification of the shape $(\mathsf{GF}\ \mathsf{true} \rightarrow \mathsf{GF}\psi)$. The point then is that, when it comes to capturing the special strong-fairness type used in FOND planning, one does not need to resort to even GR(1), and is enough to consider a less expressive type of specification.

Interestingly, the work by D'Ippolito et al. (2011, 2013) leverages on GR(1) specifications within a Software Engineering (SE) context to perform "synthesis of live controllers for fallible domains." This is indeed very related to our work and in fact inspired (Sardina & D'Ippolito, 2015), a preliminary version of this paper. There, operational live event-based models are automatically synthesized from Generalized Reactivity(1) (GR(1)) specifications of intended system behavior. GR(1) is a fragment of LTL that is both very expressive and significantly better behaved computationally for realizability and synthesis than full LTL (Bloem et al., 2012). So, the authors take GR(1) and, instead of assuming an idealized domain, as standard in the literature, explicitly model *failures* of controlled actions. They then identify a realistic fairness condition—Strong Independent Fairness—which allows for a polynomial treatment of such failures. Their SE insights and results can be imported to AI FOND planning, as we have previously done (Sardina & D'Ippolito, 2015), though only under the additional assumption that one can discriminate, at the outset, the intended effects from the failure effects of actions. This is an adequate assumption in those scenarios in which the intended effects are clear and always the same (e.g., the intended effect of a robot gripping an object is to be holding the object; a failure may involve the object slipping or even breaking). What we proposed here, in Section 4, is therefore more general and captures *full* FOND planning, which allows the same effect of an action to be an "intended" one in some cases and as "failure" in others.

We close by noting that there has been substantial work in understanding how automated planning relates to formal methods. The relation with theorem proving, for example, goes back to the work of Green (1969), and has more recently been explored by Rintanen (1999) for contingent planning in the context of more mature theorem provers (as it deals with contingent planning, fairness assumptions are not relevant). More recently, substantial interest has been shown exploring the relationship with model checking and synthesis (e.g., Camacho et al., 2016; Cimatti et al., 2003; De Giacomo, Patrizi, Felli, & Sardina, 2010; Kerjean, Kabanza, St.-Denis, & Thiébaux, 2006; Kissmann & Edelkamp, 2009; Patrizi et al., 2011, 2013; Pistore & Traverso, 2001; Pistore & Vardi, 2007; Ramirez & Sardina, 2014; Torres & Baier, 2015). Like us, Pistore and Vardi (2007) formalizes planning in LTL, but to explore variations of LTL-based goals. As in other work, their account remains at the structural properties of strong-cyclic plans, without explicit reference to the interplay between the fairness assumption and the goal, which is our focus. Their computational technique is based on tree automata with a 2EXPTIME complexity; we use Büchi games, which are arguably much simpler. In turn, De Giacomo, Patrizi, Felli, and Sardina (2010) reduces the planning task to an equivalent *efficient* (in terms of computational complexity) controller synthesis one, albeit for "standard" conditional planning (i.e., planning for strong solutions); hence, as with the work of Rintanen, the issue of fair effects does not arise.

The work of Patrizi et al. (2011; 2013), and several others, can be seen as complementary to ours. There, the authors are interested in using available planning technology to solve a limited form of LTL temporally extended goals requiring infinite plans (Patrizi et al., 2011 assumes classical deterministic domains, whereas Patrizi et al., 2013 accounts for FOND domains). They do so by encoding the deterministic Büchi automaton corresponding to the LTL goal into PDDL. The work of Camacho et al. (2016) extends that of Patrizi et al. by dropping the restriction on the LTL goals and attaining greater efficiency. While both are concerned with realizing goals beyond reachability ones using planning techniques, we are instead interested in showing how standard FOND planning (with just achievement goals) can be recast as a CTL/LTL synthesis task; so the work is complementary in some sense. Also, in essence, we are concerned with the *conceptual link* between planning and reactive synthesis, while Camacho et al.'s (2016) and Patrizi et al.'s (2011) link with reactive synthesis is only present as technical support for the actual solution to the problem. In addition, in our work, the fairness assumption is central and is not assumed built-in into the synthesis problem, but logically specified to capture the intended meaning of well-behave FOND environments. Even more, we show how to compile away such fairness requirement and perform "vanilla" synthesis on a Büchi control problem (Section 4). On the other hand, the work of Patrizi et al., as well as the recent work of Torres and Baier (2015), aim at handling LTL goals beyond standard achievement ones, but by appealing to classical (FOND) planning. This suggests that the framework we developed in this paper could be generalized too.

6. Conclusions

The overarching objective of this paper was to formally study and logically identify the kind of environments in which strong-cyclic planning is adequate, thus further completing the existing formal characterization of such type of planning in CTL (Cimatti et al., 2003; Daniele et al., 2000; Pistore & Traverso, 2001) and contributing to bridging the gap between automated planning (Ghallab, Nau, & Traverso, 2004) and reactive synthesis (Pnueli & Rosner, 1989b). To that end, in this paper, we provided three technical contributions. First, we demonstrated that an adequate fairness assumption ought to encode the *independence* of different non-deterministic steps, and developed a characterization of such environments in CTL*/LTL that does so. Secondly, and more importantly, we proved that strong-cyclic policies, as defined in the literature are sound and complete solution concepts for such type of environments. Whereas this has always been given as granted in the field, this is, to our knowledge, the first work that crystallizes it formally and in the same foundational framework as that of Daniele et al.. We then argued that the logical characterization obtained, allows, in principle, for direct specification of the FOND planning task into a reactive synthesis one, albeit one that appears not amenable for efficient computation. Lastly, we demonstrated that the particular type of (strong) fairness required can be carefully encoded into a Büchi game fairness type in an efficient manner, thus achieving the EXPTIME lower bound complexity of non-deterministic planning. This generalizes our preliminary result (Sardina & D'Ippolito, 2015) that showed that, for the special case of FOND problems with explicit "intended effects" for actions, low complexity synthesis techniques were applicable.

It is important to stress that, with these results, we do not intend to claim that controller synthesis is, today, as effective in practice as existing optimized FOND system, such as the efficient PRP (Muise et al., 2012) planner. For one, standard out-of-the-box reactive synthesis approaches will generally perform full exploration of the state space and yield sort of universal solution policies, whereas state-of-the-art FOND planners often aim (and succeed) at avoiding full exploration of the state space (though outputting one single solution policy). Regardless, experimental evaluation is out of the scope of the current paper. What the results do intend is to further crystallize what FOND planning is and support cross-fertilization between the two synthesis areas. We believe that making the connection between planning and controller synthesis more evident can benefit both areas. Automated planning can exploit recent powerful techniques—like synthesis of GR(1) or safety specifications—and open for more general planning settings. As pointed out by De Giacomo, Patrizi, and Sardina (2010), one could be interested in planning under "selective" and/or "conditional" fairness assumptions (e.g., the dice is fair unless an action has "loaded" it), or planning under actions with "intended" and "failure" effects (D'Ippolito et al., 2011). The recent work by Camacho and McIlraith (2016) explores FOND planning in the context of more refined description of action transitions under which the fairness assumption may not necessarily hold. For reactive synthesis, planning settings can provide concrete applications and inform what types of system goal specifications are meaningful and worth studying. Also, several promising techniques and systems have emerged for solving FOND planning problems, which, as we have noted in the previous section may inform alternative computational approaches for controller synthesis.

In this article, we have restricted our main results, such as Theorem 3 and Corollary 1, to *deterministic* policies. It turns out that, for *non-deterministic* policies—those prescribing more than one action in a state—the assumption on state-strong fair on the environment (Definition 6), is not sufficient. To see that, consider a state s and $\pi(s) = \{a, b\}$ (in state s, policy π prescribes the execution of a or b). Suppose that action a has no effects and hence it behaves like a no-op action in s (i.e., when executed in s the unique successor state is s itself). In turn, action b is non-deterministic and may transition s to either s itself or to goal state s_g. It is not hard to see that π does meet Definition 5 and is hence strong-cyclic: the goal is always reachable from s. However, it is not enough for the environment to behave in a fair manner if the executor of π actually chooses to execute action a always! Indeed, there are executions of policy π from state s in state-strong fair environments that never achieve the goal. To recover the technical results presented in this work, one also needs to require the policy to be carried out by a fair executor, that is, one that chooses every prescribed action in each state infinitely often (this can be captured with a similar version of Definition 6). Also, we note that it is not the case that any deterministic (naive) projection of a non-deterministic strong-cyclic policy is also a strong-cyclic. In the example above, projection $\pi'(s) = \{a\}$ is not strong-cyclic anymore. Nonetheless, it is not difficult to see that the strong-cyclic deterministic policies encoded inside a non-deterministic one can be checked or extracted in efficiently—in polynomial time—using a backward (from the goal) type of algorithm.

From the work presented in this article, there are many interesting possible research lines to pursue. We plan, for example, to investigate other solution concepts for non-deterministic planning. In particular, we are interested in fault-tolerant planning (Domshlak, 2013) and seek solution plans that may accept failures in certain circumstances. Going one step beyond, it would be interesting to develop a framework for FOND planning with hybrid type of solution concepts, integrating standard strong-cyclic with strong, fault-tolerant, and even weak sub-plans, and under conditional fairness assumptions. This poses both representation as well as computational challenges that, we believe, are worth exploring at this stage.

Acknowledgments

We thank the anonymous reviewers (of this article and our previous version at IJCAI'15), for their comments and acknowledge the support of the Australian Research Council under a Discovery Project (DP120100332), and projects ANPCYT PICT 2012-0724/2011-1774/2013-2341, UBACYT 036/0384, CONICET PIP 11220110100596CO, and MEALS 295261.

References

Abadi, M., Lamport, L., & Wolper, P. (1989). Realizable and unrealizable specifications of reactive systems. In *Proceedings of the international colloquium on automata, languages and programming (ICALP)* (pp. 1–17).

Baier, C., & Katoen, J. (2008). *Principles of model checking*. The MIT Press.

Bloem, R., Jobstmann, B., Piterman, N., Pnueli, A., & Sa'ar, Y. (2012). Synthesis of reactive(1) designs. *Journal of Computer and System Sciences*, *78*(3), 911–938.

Bryce, D., & Buffet, O. (2008). The 6th international planning competition: Uncertainty track. In *Proceedings of international planning competition (IPC).*

Buchi, J., & Landweber, L. (1969). Solving sequential conditions by finite-state strategies. *Transactions of the American Mathematical Society*, 295–311.

Bylander, T. (1994). The computational complexity of propositional strips planning. *Artificial Intelligence*, *69*, 165–204.

Camacho, A., & McIlraith, S. A. (2016). Strong-cyclic planning when fairness is not a valid assumption. In *Proceedings of the workshop on knowledge-based techniques for problem solving and reasoning.* (`http://ktiml.mff.cuni.cz/ bartak/KnowProS2016`)

Camacho, A., Triantafillou, E., Muise, C. J., Baier, J. A., & McIlraith, S. A. (2016). Nondeterministic planning with temporally extended goals: Completing the story for finite and infinite LTL. In *Proceedings of the workshop on knowledge-based techniques for problem solving and reasoning.* (`http://ktiml.mff.cuni.cz/ bartak/KnowProS2016`)

Chatterjee, K., Jurdzinski, M., & Henzinger, T. A. (2004). Quantitative stochastic parity games. In *Proceedings of the annual ACM-SIAM symposium on discrete algorithms (SODA)* (pp. 121–130).

Church, A. (1963). Logic, arithmetic, and automata. In *Proceedings of the international congress of mathematicians* (pp. 23–35).

Cimatti, A., Pistore, M., Roveri, M., & Traverso, P. (2003). Weak, strong, and strong cyclic planning via symbolic model checking. *Artificial Intelligence*, *147*(1-2), 35–84.

Ciolek, D., Braberman, V. A., D'Ippolito, N., & Uchitel, S. (2016). Directed controller synthesis of discrete event systems: Taming composition with heuristics. In *Proeedings of the IEEE conference on decision and control (CDC)* (pp. 4764–4769).

Clarke, E., & Emerson, E. (1982). Design and synthesis of synchronization skeletons using branching time temporal logic. In D. Kozen (Ed.), *Logics of programs* (Vol. 131, pp. 52–71).

Springer.

Clarke, E., Grumberg, O., & Peled, D. (1999). *Model checking*. Springer.

Daniele, M., Traverso, P., & Vardi, M. (2000). Strong cyclic planning revisited. *Recent Advances in AI Planning*, 35–48.

De Giacomo, G., Patrizi, F., Felli, P., & Sardina, S. (2010). Two-player game structures for generalized planning and agent composition. In *Proceedings of the national conference on artificial intelligence (AAAI)* (pp. 297–302).

De Giacomo, G., Patrizi, F., & Sardina, S. (2010). Generalized planning with loops under strong fairness constraints. In *Proceedings of the international conference on principles of knowledge representation and reasoning (KR)* (pp. 351–361).

De Giacomo, G., & Vardi, M. Y. (2015). Synthesis for LTL and LDL on finite traces. In *Proceedings of the international joint conference on artificial intelligence (IJCAI)* (pp. 1558–1564).

D'Ippolito, N., Braberman, V. A., Piterman, N., & Uchitel, S. (2011). Synthesis of live behaviour models for fallible domains. In *Proceedings of the international conference on software engineering (ICSE)* (pp. 211–220).

D'Ippolito, N., Braberman, V. A., Piterman, N., & Uchitel, S. (2013). Synthesizing non-anomalous event-based controllers for liveness goals. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *22*(1), 9:1–9:36.

Domshlak, C. (2013). Fault tolerant planning: Complexity and compilation. In *Proceedings of the international conference on automated planning and scheduling (ICAPS)* (pp. 64–72).

Emerson, E. A. (1990). Temporal and modal logic. In *Handbook of theoretical computer science, volume B: Formal models and sematics* (Vol. B, pp. 995–1072). The MIT Press.

Emerson, E. A., & Halpern, J. Y. (1986). "Sometimes" and "not never" revisited: On branching versus linear time temporal logic. *Journal of the ACM*, *33*(1), 151–178.

Forejt, V., Kwiatkowska, M. Z., Norman, G., & Parker, D. (2011). Automated verification techniques for probabilistic systems. In *Proceedings of the international school on formal methods for the design of computer (SFM)* (pp. 53–113).

Fu, J., Ng, V., Bastani, F., & Yen, I.-L. (2011). Simple and fast strong cyclic planning for fully-observable non-deterministic planning problems. In *Proceedings of the international joint conference on artificial intelligence (IJCAI)* (pp. 1949–1954).

Geffner, H., & Bonet, B. (2013). *A concise introduction to models and methods for automated planning*. Morgan & Claypool Publishers.

Gerevini, A., Bonet, B., & Givan, B. (Eds.). (2006). *Booklet of 4th international planning competition.* Retrieved from `http://www.ldc.usb.ve/~bonet/ipc5/`

Ghallab, M., Nau, D. S., & Traverso, P. (2004). *Automated planning: Theory and practice*. Morgan Kaufmann Publishers Inc.

Green, C. (1969). Application of theorem-proving to problem solving. In *Proceedings of the international joint conference on artificial intelligence (IJCAI)* (pp. 219–239).

Kerjean, S., Kabanza, F., St.-Denis, R., & Thiébaux, S. (2006). Analyzing LTL model checking techniques for plan synthesis and controller synthesis (work in progress). *Electronic Notes in Theoretical Computer Science (ENTCS)*, *149*(2), 91-104.

Kesten, Y., Piterman, N., & Pnueli, A. (2005, July). Bridging the gap between fair simulation and trace inclusion. *Journal Information and Computation*, *200*, 35–61.

Kissmann, P., & Edelkamp, S. (2009). Solving fully-observable non-deterministic planning problems via translation into a general game. In *Proceedings of the annual german conference on AI* (pp. 1–8).

Kupferman, O., Piterman, N., & Vardi, M. Y. (2006). Safraless compositional synthesis. In *Proceedings of the international conference on computer aided verification (CAV)* (pp. 31–44).

Kuter, U., Nau, S., D., Reisner, E., & Goldman, P., R. (2008). Using classical planners to solve nondeterministic planning problems. In *Proceedings of the international conference on automated planning and scheduling (ICAPS)* (pp. 190–197).

Maler, O., Pnueli, A., & Sifakis, J. (1995). On the synthesis of discrete controllers for timed systems. In *Stacs 95* (pp. 229–242).

Manna, Z., & Waldinger, R. (1987). How to clear a block: A theory of plans. *Journal of Automed Reasoning*, *4*(3), 343–377.

Martin, D. (1975). Borel determinacy. *Annals of Mathematics*, 363–371.

Muise, C., Belle, V., & McIlraith, S. A. (2014). Computing contingent plans via fully observable non-deterministic planning. In *Proceedings of the national conference on artificial intelligence (AAAI)* (pp. 2322–2329).

Muise, C., McIlraith, S. A., & Beck, J. C. (2012). Improved non-deterministic planning by exploiting state relevance. In *Proceedings of the international conference on automated planning and scheduling (ICAPS)* (pp. 172–180).

Muise, C., McIlraith, S. A., & Belle, V. (2014). Non-deterministic planning with conditional effects. In *Proceedings of the international conference on automated planning and scheduling (ICAPS)* (pp. 370–374).

Ortlieb, M., & Mattmüller, R. (2013). Pattern-database heuristics for partially observable nondeterministic planning. In *Proceedings of the annual german conference on AI* (pp. 140–151). doi: 10.1007/978-3-642-40942-4_13

Patrizi, F., Lipovetzky, N., De Giacomo, G., & Geffner, H. (2011). Computing infinite plans for LTL goals using a classical planner. In *Proceedings of the international joint conference on artificial intelligence (IJCAI)* (pp. 2003–2008).

Patrizi, F., Lipovetzky, N., & Geffner, H. (2013). Fair LTL synthesis for non-deterministic systems using strong cyclic planners. In *Proceedings of the international joint conference on artificial intelligence (IJCAI).*

Pistore, M., & Traverso, P. (2001). Planning as model checking for extended goals in non-deterministic domains. In *Proceedings of the international joint conference on artificial intelligence (IJCAI)* (pp. 479–486).

Pistore, M., & Vardi, M. Y. (2007). The planning spectrum - one, two, three, infinity. *Journal of Artificial Intelligence Research (JAIR)*, *30*, 101–132.

Piterman, N., Pnueli, A., & Sa'ar, Y. (2006). Synthesis of reactive(1) designs. In *Proceedings of the international conference on verification, model checking, and abstract interpretation (VMCAI)* (pp. 364–380).

Pnueli, A. (1977). The temporal logic of programs. In *Proceedings of the annual symposium on foundations of computer science (SFCS)* (pp. 46–57).

Pnueli, A., & Rosner, R. (1989a). On the synthesis of an asynchronous reactive module. In *Proceedings of the international colloquium on automata, languages and programming (ICALP)* (pp. 652–671).

Pnueli, A., & Rosner, R. (1989b). On the synthesis of a reactive module. In *Proceedings of the ACM SIGPLAN-SIGACT symposium on principles of programming languages (POPL)* (pp. 179–190).

Ramirez, M., & Sardina, S. (2014). Directed fixed-point regression-based planning for non-deterministic domains. In *Proceedings of the international conference on automated planning and scheduling (ICAPS)* (pp. 235–243).

Rintanen, J. (1999). Constructing conditional plans by a theorem-prover. *Journal of Artificial Intelligence Research (JAIR)*, *10*, 323–352.

Rintanen, J. (2003). Expressive equivalence of formalisms for planning with sensing. In *Proceedings of the international conference on automated planning and scheduling (ICAPS)* (pp. 185–194).

Rintanen, J. (2004). Complexity of planning with partial observability. In *Proceedings of the international conference on automated planning and scheduling (ICAPS)* (pp. 345–354).

Rintanen, J. (2008). Regression for classical and nondeterministic planning. In *Proceedings of the european conference in artificial intelligence (ECAI)* (pp. 568–572).

Rosner, R. (1992). *Modular synthesis of reactive systems* (Unpublished doctoral dissertation). Weizmann Institute of Science.

Sardina, S., & D'Ippolito, N. (2015). Towards fully observable non-deterministic planning as assumption-based reactive synthesis. In *Proceedings of the international joint conference on artificial intelligence (IJCAI)* (pp. 3200–3206).

Torres, J., & Baier, J. A. (2015). Polynomial-time reformulations of LTL temporally extended goals into final-state goals. In *Proceedings of the international joint conference on artificial intelligence (IJCAI)* (pp. 1696–1703).

Vardi, M. Y. (1995). An automata-theoretic approach to fair realizability and synthesis. In *Proceedings of the international conference on computer aided verification (CAV)* (pp. 267–278).

Vardi, M. Y. (1996). An automata-theoretic approach to linear temporal logic. In *Logics for concurrency: Structure versus automata* (Vol. 1043, p. 238-266). Springer.

von Neumann, J., Morgenstern, O., Kuhn, H., & Rubinstein, A. (1944). *Theory of games and economic behavior (60th anniversary commemorative edition)*. Princeton University Press.